

**КИЇВСЬКИЙ СТОЛИЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ БОРИСА ГРІНЧЕНКА**

Факультет інформаційних технологій та математики
Кафедра інформаційної та кібернетичної безпеки імені професора
Володимира Бурячка

Гринкевич Г.О., Василенко В.В., Рудін Д.Г.

**МОНІТОРИНГ, АУДИТ ТА
АДМІНІСТРУВАННЯ ЗАХИЩЕНИХ
ІТ-СИСТЕМ І МЕРЕЖ**

Підручник

Київ 2026

УДК 004.056:004.7(075.8)

Рекомендовано Вченою радою Факультету інформаційних технологій та математики Київського столичного університету імені Бориса Грінченка (протокол № 9 від 16 вересня 2026 р.)

Автори:

Гринкевич Г.О. — доктор технічних наук, доцент, професор кафедри інформаційної та кібернетичної безпеки імені професора Володимира Бурячка Київського столичного університету імені Бориса Грінченка;

Василенко В.В. — кандидат технічних наук, доцент, доцент кафедри комп'ютерних наук Державного університету інформаційно-комунікаційних технологій;

Рудін Д.Г. — Platform Engineer, незалежний дослідник.

Рецензенти:

Опірський Іван Романович, доктор технічних наук, професор, завідувач кафедри захисту інформації Національного університету «Львівська політехніка»

Мамченко Сергій Миколайович, доктор педагогічних наук, професор, професор кафедри комп'ютерних систем, мереж та кібербезпеки Національного університету біоресурсів та природокористування України

Гринкевич Г.О. Моніторинг, аудит та адміністрування захищених ІТ-систем і мереж: підручник / Г.О. Гринкевич, В.В. Василенко, Д.Г. Рудін. – Київ : Київський столичний університет імені Бориса Грінченка, 2026. – 256 с.

У підручнику системно викладено теоретичні та практичні засади моніторингу, адміністрування та аудиту захищених інформаційно-телекомунікаційних систем і мереж відповідно до робочої програми однойменної навчальної дисципліни для магістрів спеціальності 125 «Кібербезпека». Матеріал структуровано у трьох модулях: моніторинг (журналювання, SIEM, Microsoft Sentinel, Splunk, аналіз мережевого трафіку, DLP, Threat Intelligence), адміністрування (політика безпеки, захист мереж, IDS/IPS, побудова мереж за моделлю OSI, хмарна інфраструктура, Kubernetes) та аудит (стандарти ISO 19011/ISO 27001/NIST, процедури аудиту, аудит цілісності файлових систем,

протидія аудиту, AI-аналіз подій безпеки на основі машинного навчання та UEBA). Кожен розділ супроводжується практикумом або лабораторною роботою, контрольними питаннями; усі джерела зведено в єдиний список наприкінці підручника. Підручник призначений для студентів і викладачів закладів вищої освіти, а також практикуючих фахівців із кібербезпеки, зацікавлених у системному викладі сучасних підходів до побудови центрів моніторингу безпеки (SOC).

Матеріал підручника побудовано за принципом послідовного поєднання офіційних тем робочої програми з їх сучасним технологічним наповненням: кожна тема, вже закладена в затверджену програму дисципліни, розкривається не лише на рівні класичних понять, а й доповнюється актуальним інструментарієм, який реально застосовується в сучасних центрах моніторингу безпеки. Такий підхід дозволяє використовувати підручник як самостійний навчально-методичний ресурс, що не потребує постійного звернення до розрізнених джерел, і водночас зберігає прямий зв'язок із затвердженою структурою дисципліни, її кредитним навантаженням та розподілом аудиторних і самостійних годин.

Модуль 1 присвячено моніторингу як безперервному процесу спостереження за станом захищеності інформаційної системи: від базових понять і протоколу SNMP до архітектури сучасних SIEM-рішень, платформи Microsoft Sentinel, системи Splunk, методів аналізу мережевого трафіку, технологій запобігання витоку даних (DLP) та інтеграції розвідки кіберзагроз (Threat Intelligence) у загальний контур захисту. Модуль 2 розкриває адміністрування захищених інформаційно-телекомунікаційних систем: ієрархію документів політики безпеки, управління змінами та вразливостями, побудову захищених мереж відповідно до моделі OSI, системи виявлення та запобігання вторгненням (IDS/IPS), а також сучасні аспекти адміністрування хмарної інфраструктури та контейнеризованих середовищ на основі Kubernetes. Модуль 3 завершує коло розгляду аудитом як незалежною, ретроспективною перевіркою відповідності системи встановленим вимогам: методологію та стандарти аудиту інформаційної безпеки (ISO 19011, ISO/IEC 27001, NIST SP 800-53A), практичні процедури збору доказів і формування звітності, аудит цілісності файлових систем, дослідження засобів протидії аудиту та застосування штучного інтелекту й машинного навчання для аналізу подій безпеки в SIEM/SOC.

Кожен із двадцяти чотирьох розділів підручника має єдину структуру: тематичний вступ, послідовний виклад підрозділів з оригінальними ілюстраціями та таблицями, практикум або лабораторну роботу для закріплення практичних навичок, розширений перелік контрольних питань (не менше п'ятнадцяти на розділ).

Довідковий апарат видання доповнено глосарієм термінів, зведеним переліком використаних стандартів, словником аббревіатур, узагальненим переліком контрольних питань за модулями та єдиним зведеним списком використаних джерел, що включає фахову літературу, профільні стандарти й опубліковані наукові праці авторів підручника та спрощує підготовку студентів до підсумкового контролю.

Підручник призначений передусім для студентів другого (магістерського) рівня вищої освіти спеціальності 125 «Кібербезпека», однак його практична спрямованість і системний виклад роблять його корисним також для викладачів суміжних дисциплін, аспірантів і практикуючих фахівців із кібербезпеки, зокрема аналітиків центрів моніторингу безпеки (SOC), адміністраторів захищених інформаційних систем та внутрішніх аудиторів інформаційної безпеки, які прагнуть системно впорядкувати практичний досвід у межах єдиної методологічної рамки.

УДК 004.056:004.7(075.8)

© Гринкевич Г.О., Василенко В.В., Рудін Д.Г., 2026
© Київський столичний університет імені Бориса Грінченка, 2026

Зміст

МОДУЛЬ 1. МОНІТОРИНГ ЗАХИЩЕНИХ ІТ-СИСТЕМ І МЕРЕЖ ... 13

РОЗДІЛ 1. ОСНОВИ МОНІТОРИНГУ ЗАХИЩЕНИХ ІТ-СИСТЕМ І МЕРЕЖ 13

1.1. Поняття, мета та завдання моніторингу інформаційної безпеки	14
1.2. Об'єкти, суб'єкти та рівні моніторингу	16
1.3. Нормативно-правові засади моніторингу інформаційної безпеки	17
1.4. Метрики та показники ефективності моніторингу	17
1.5. Моніторинг мережевого обладнання за протоколом SNMP	18
1.6. Активна перевірка доступності служб: ICMP та синтетичний моніторинг	20
1.7. Побудова базової лінії нормальної поведінки системи (baseline)	21
1.8. Платформи мережевого моніторингу: огляд практичних інструментів	21
1.9. Виклики масштабування системи моніторингу у великих мережах	22
1.10. Моніторинг як послуга: власний SOC проти аутсорсингу (MSSP)	23
1.11. Інтеграція моніторингу із системою обліку заявок (ITSM)	23
Практикум 1. Розгортання сканера вразливостей і базове сканування мережевого сегмента	25
Контрольні питання до розділу 1	25

РОЗДІЛ 2. ЖУРНАЛЮВАННЯ ТА ЦЕНТРАЛІЗОВАНИЙ ЗБІР ПОДІЙ БЕЗПЕКИ

.....	26
2.1. Роль журналювання в забезпеченні кібербезпеки	27
2.2. Джерела подій безпеки	28
2.3. Формати та протоколи журналювання	28
2.4. Архітектура конвеєра централізованого збору подій	29
2.5. Нормалізація та збагачення подій	30
2.6. Політики зберігання та забезпечення цілісності журналів	30
2.7. Журналювання в хмарному та контейнеризованому середовищі	31
2.8. Управління обсягом журналів: багаторівневе зберігання та фільтрація малоцінних подій	32
.....	32
2.9. Поглиблене журналювання кінцевих точок: за межами стандартного журналу ОС	32
2.10. Журнал як судовий доказ: цілісність ланцюга зберігання	33
2.11. Стиснення та архівування історичних журналів	34
Практикум 2. Побудова конвеєра централізованого збору подій	35
Контрольні питання до розділу 2	35

РОЗДІЛ 3. SIEM-СИСТЕМИ: АРХІТЕКТУРА, ФУНКЦІЇ, ОГЛЯД РИНКУ РІШЕНЬ

.....	36
3.1. Визначення SIEM та еволюція технології	37
3.2. Архітектура SIEM	38
3.3. Методи кореляції подій	38
3.4. Огляд ринку рішень	40
3.5. Критерії вибору SIEM для організації	40
3.6. Sigma: універсальна мова опису правил виявлення	41

3.7. UEBA: аналітика поведінки користувачів і сутностей	41
3.8. Тюнінг правил кореляції та боротьба з втомою від сповіщень	42
3.9. SIEM і відповідність регуляторним вимогам: звітність compliance	43
3.10. SIEM у гібридному й мультихмарному середовищі	43
Практикум 3. Порівняльний аналіз SIEM-рішень	44
Контрольні питання до розділу 3	44

РОЗДІЛ 4. MICROSOFT SENTINEL: РОЗГОРТАННЯ ТА АДМІНІСТРУВАННЯ 45

4.1. Місце Sentinel в екосистемі безпеки Microsoft	46
4.2. Основні компоненти Sentinel	46
4.3. Основи мови запитів KQL (Kusto Query Language)	47
4.4. Підключення джерел даних	48
4.5. Аналітичні правила та автоматизація реагування	48
4.6. Керування вартістю: рівні збереження даних	49
4.7. UEBA та інтеграція Threat Intelligence в Sentinel	49
4.8. Workbooks: візуалізація та інтерактивна звітність	50
4.9. Керування доступом і багатоклієнтська архітектура Sentinel	50
4.10. Content Hub і готові рішення для типових сценаріїв виявлення	51
Практикум 4. Розгортання робочої області Sentinel і базового правила кореляції	51
Контрольні питання до розділу 4	52

РОЗДІЛ 5. SPLUNK: АРХІТЕКТУРА, ПОШУК ПОДІЙ, ДАШБОРДИ.....52

5.1. Архітектура Splunk	53
5.2. Дані у Splunk: індекси, sourcetype, поля	54
5.3. Мова пошуку SPL (Search Processing Language)	55
5.4. Common Information Model і Splunk Enterprise Security	55
5.5. Дашборди та візуалізація	56
5.6. Ліцензування та управління обсягом даних	56
5.7. Splunk SOAR: автоматизація реагування	56
5.8. Кластеризація та відмовостійкість Splunk	57
5.9. Керування доступом і ролями у Splunk	57
5.10. Machine Learning Toolkit: вбудовані засоби аналітики Splunk	58
Практикум 5. Побудова дашборду моніторингу автентифікацій у Splunk	59
Контрольні питання до розділу 5	59

РОЗДІЛ 6. ВИЯВЛЕННЯ МЕРЕЖЕВИХ АТАК ШЛЯХОМ АНАЛІЗУ ТРАФІКУ.60

6.1. Методи аналізу мережевого трафіку	61
6.2. Системи виявлення вторгнень мережевого рівня	62
6.3. Типові мережеві атаки та їхні ознаки в трафіку	62
6.4. Моніторинг за допомогою NetFlow/IPFIX	63
6.5. Аналіз зашифрованого трафіку без розшифрування	63
6.6. Виявлення й пом'якшення DDoS-атак	64
6.7. Honeypot та обманні технології (deception technology)	64
6.8. Виявлення латерального руху за допомогою аналізу потоків	65
6.9. DNS-моніторинг як індикатор компрометації	65
Практикум 6. Аналіз мережевого трафіку для виявлення атак	67
Контрольні питання до розділу 6	67

РОЗДІЛ 7. DLP-СИСТЕМИ: ОСНОВИ ЗАСТОСУВАННЯ	68
7.1. Призначення та архітектура DLP-систем.....	69
7.2. Методи детекції контенту	70
7.3. Канали контролю	71
7.4. Огляд SearchInform DLP	71
7.5. Розробка політик DLP і юридичні аспекти моніторингу співробітників.....	72
7.6. DLP у хмарному середовищі та для віддалених працівників	72
7.7. Метрики ефективності програми DLP	73
7.8. Класифікація даних як передумова ефективної DLP	73
7.9. Виявлення інсайдерських загроз: поєднання DLP і поведінкової аналітики	74
7.10. Контроль хмарних застосунків: CASB як доповнення до традиційної DLP	74
Практикум 7. Налаштування політики DLP	76
Лабораторна робота. Встановлення та налагодження програмного комплексу SearchInform	76
Контрольні питання до розділу 7	76
РОЗДІЛ 8. THREAT INTELLIGENCE: ЗБІР, АНАЛІЗ, ІНТЕГРАЦІЯ З SIEM	77
8.1. Поняття та рівні Threat Intelligence	78
8.2. Життєвий цикл кіберрозвідки (Intelligence Cycle)	78
8.3. Джерела Threat Intelligence	80
8.4. Стандарти обміну та класифікації	80
8.5. Платформи Threat Intelligence	80
8.6. Інтеграція Threat Intelligence з SIEM	81
8.7. Оцінювання якості та достовірності Threat Intelligence.....	81
8.8. Threat Hunting: проактивне використання Threat Intelligence	82
8.9. Атрибуція атак: угруповання загроз та їхні профілі.....	82
8.10. Обмін розвідданими на національному й галузевому рівні в Україні	83
8.11. MITRE ATT&CK як спільна мова Threat Intelligence і SOC	83
Практикум 8. Імпорт індикаторів компрометації та побудова правила збагачення.....	85
Контрольні питання до розділу 8.....	85
МОДУЛЬ 2. АДМІНІСТРУВАННЯ ЗАХИЩЕНИХ ІТ-СИСТЕМ І МЕРЕЖ.....	87
РОЗДІЛ 9. АДМІНІСТРУВАННЯ ІНФОРМАЦІЙНИХ СИСТЕМ І ПОЛІТИКА БЕЗПЕКИ ІТС.....	87
9.1. Поняття та завдання адміністрування інформаційної системи.....	87
9.2. Структура документів політики інформаційної безпеки.....	88
9.3. Життєвий цикл адміністрування ІТС	90
9.4. Ролі, обов'язки та принцип найменших привілеїв.....	91
9.5. Управління змінами та документування конфігурацій	91
9.6. Управління вразливостями (Vulnerability Management)	92
9.7. Управління активами ІТ та конфігураційна база даних (CMDB).....	92
9.8. Патч-менеджмент: тестування, вікна обслуговування, відкат.....	93
9.9. Документація адміністратора та бази знань (runbooks).....	93
9.10. Еталонні базові профілі конфігурації (CIS Benchmarks).....	94
Практикум 9. Складання фрагмента процедури управління змінами.....	96

Контрольні питання до розділу 9	96
РОЗДІЛ 10. ЗАХИСТ ІНФОРМАЦІЇ В КОМП'ЮТЕРНИХ МЕРЕЖАХ. ФУНКЦІЇ АДМІНІСТРАТОРА БЕЗПЕКИ	97
10.1. Функції та повноваження адміністратора безпеки.....	98
10.2. Багаторівневий захист мережі (Defense in Depth)	98
10.3. Керування доступом: автентифікація, авторизація, аудит (AAA).....	99
10.4. Сегментація мережі та міжмережеві екрани.....	100
10.5. Роль адміністратора безпеки в реагуванні на інциденти.....	101
10.6. Віддалений доступ: VPN та засади Zero Trust Network Access.....	101
10.7. Управління привілейованим доступом (Privileged Access Management, PAM)	102
10.8. Аудит і оптимізація правил міжмережевого екрана	102
10.9. Інфраструктура відкритих ключів (PKI) і керування сертифікатами.....	103
10.10. Контроль доступу до мережі на каналному рівні (NAC, 802.1X).....	104
Практикум 10. Розроблення фрагмента політики паролів та мережевої сегментації.....	105
Контрольні питання до розділу 10.....	105
РОЗДІЛ 11. АДМІНІСТРУВАННЯ БАЗОВИХ ІНФОРМАЦІЙНИХ СЛУЖБ	106
11.1. Служба каталогів та контролер домену	107
11.2. Служба DNS: адміністрування та захист	108
11.3. Служба DHCP: адміністрування та типові атаки	108
11.4. Групові політики та централізоване управління конфігурацією	109
11.5. Резервне копіювання та відновлення служб	109
11.6. Автентифікація домену електронної пошти: SPF, DKIM, DMARC	110
11.7. Синхронізація часу: служба NTP та її роль у безпеці.....	110
11.8. Веб-проксі та фільтрація вихідного вебтрафіку	111
11.9. Файлові служби та служби друку: адміністрування спільних ресурсів.....	112
11.10. Основи захисту серверів баз даних.....	112
Лабораторна робота 11. Розгортання контрольованого середовища DNS/DHCP та аналіз атаки rogue DHCP.....	113
Контрольні питання до розділу 11	113
РОЗДІЛ 12. СИСТЕМИ ВИЯВЛЕННЯ ВТОРГНЕНЬ (IDS/IPS): АДМІНІСТРУВАННЯ.....	114
12.1. Класифікація систем виявлення вторгнень.....	115
12.2. Розміщення IDS та IPS у мережі	115
12.3. Методи виявлення: сигнатурні, аномалій, гібридні	116
12.4. Огляд поширених рішень з відкритим кодом.....	117
12.5. Адміністрування правил і зниження частки хибних спрацювань	117
12.6. Методи ухилення від систем виявлення вторгнень	118
12.7. Продуктивність і відмовостійкість IPS-рішень	118
12.8. EDR/XDR: виявлення на кінцевій точці як доповнення до мережевого IDS/IPS	119
12.9. Перевірка ефективності виявлення: адверсарна симуляція та purple team.....	120
12.10. NDR (Network Detection and Response) як доповнення до IDS/IPS	120
Практикум 12. Написання та тестування власного правила Suricata	122
Контрольні питання до розділу 12.....	122

РОЗДІЛ 13. ПОБУДОВА ЗАХИЩЕНИХ МЕРЕЖ ВІДПОВІДНО ДО МОДЕЛІ OSI	123
13.1. Модель OSI як основа аналізу захисту мережі	124
13.2. Захист на фізичному, каналному та мережевому рівнях	125
13.3. Програмно-конфігуровані мережі (SDN) як сучасний підхід до побудови захищених мереж	126
13.4. Захист на транспортному, сеансовому та прикладному рівнях	127
13.5. TLS у деталях: рукописання та перехід від версії 1.2 до 1.3	127
13.6. Мікросегментація мережі на основі SDN	128
13.7. Захист бездротових мереж (Wi-Fi)	129
13.8. VLAN і тунелінг: технічна реалізація сегментації каналного рівня	130
13.9. Мережева документація: схеми та реєстр IP-адрес (IPAM)	130
Практикум 13. Побудова матриці засобів захисту для багаторівневої мережі	131
Контрольні питання до розділу 13	131
РОЗДІЛ 14. ХМАРНА ІНФРАСТРУКТУРА: МОДЕЛІ, СПІЛЬНА ВІДПОВІДАЛЬНІСТЬ, БЕЗПЕКА, АДМІНІСТРУВАННЯ	132
14.1. Моделі хмарних послуг: IaaS, PaaS, SaaS	133
14.2. Моделі розгортання хмари	133
14.3. Модель спільної відповідальності за безпеку	134
14.4. Infrastructure as Code: концепція та інструменти	135
14.5. Відмовостійкість і аварійне відновлення хмарних систем	136
14.6. Адміністрування хмарних ресурсів	137
14.7. Захист хмарних сховищ даних	137
14.8. Безпека безсерверних обчислень (Serverless/FaaS)	138
14.9. Мережева безпека в хмарі: групи безпеки та віртуальні мережі	138
14.10. Гібридна зв'язність: з'єднання локальної інфраструктури з хмарою	139
14.11. FinOps: контроль витрат як складова адміністрування хмари	140
Практикум 14. Опис навчального хмарного середовища у форматі Infrastructure as Code	141
Контрольні питання до розділу 14	141
РОЗДІЛ 15. КОНТЕЙНЕРИЗАЦІЯ ТА KUBERNETES: БЕЗПЕКА І АДМІНІСТРУВАННЯ	142
15.1. Контейнери проти віртуальних машин	143
15.2. Docker: архітектура та основні поняття	144
15.3. Архітектура кластера Kubernetes	144
15.4. Безперервна інтеграція та доставка контейнеризованих сервісів	145
15.5. Безпека контейнерів і Kubernetes	146
15.6. Секрети та конфігурація в Kubernetes	147
15.7. Service Mesh: взаємна автентифікація та безпека трафіку між мікросервісами	147
15.8. Спостережуваність контейнеризованих застосунків	148
15.9. Управління ресурсами та автоматичне масштабування	149
15.10. GitOps: декларативне управління конфігурацією Kubernetes	149
Практикум 15. Розгортання застосунку в Kubernetes із застосуванням RBAC та мережевої політики	151
Контрольні питання до розділу 15	151

РОЗДІЛ 16. ПРАКТИКУМ: НАЛАШТУВАННЯ SEARCHINFORM ДЛЯ КОНТРОЛЮ МЕРЕЖІ	152
16.1. Призначення та архітектура SearchInform	153
16.2. Налаштування політики контролю мережевих каналів	154
16.3. Розслідування інцидентів за допомогою повнотекстового індексу	154
16.4. Інтеграція SearchInform з іншими засобами моніторингу організації	155
16.5. Продуктивність і масштабування SearchInform у великих організаціях	156
16.6. Типові причини невдалого впровадження DLP	156
16.7. Огляд альтернативних рішень на ринку DLP	157
16.8. Юридичний супровід впровадження DLP: узгодження з профспілкою/трудовим колективом	157
Практикум 16. Розгортання політики контролю витоку конфіденційних документів	158
Контрольні питання до розділу 16	159

МОДУЛЬ 3. АУДИТ ЗАХИЩЕНИХ ІТ-СИСТЕМ І МЕРЕЖ161

РОЗДІЛ 17. ОСНОВНІ ПІДХОДИ ДО МОНІТОРИНГУ ТА АУДИТУ БЕЗПЕКИ ІНФОРМАЦІЇ	161
17.1. Аудит інформаційної безпеки: визначення й місце в циклі забезпечення безпеки	162
17.2. Види аудиту: внутрішній, зовнішній, сертифікаційний	162
17.3. Модель трьох ліній захисту (Three Lines Model)	163
17.4. Об'єкти аудиту в захищеній ІТС	164
17.5. Принципи проведення аудиту за ISO 19011	165
17.6. Ролі учасників аудиту	165
17.7. Аудит, комплаєнс і тестування на проникнення: розмежування суміжних понять	166
17.8. Програма аудиту організації: багаторічне планування	166
17.9. Ризик-орієнтоване планування аудиту: пріоритизація об'єктів за рівнем ризику	167
Практикум 17. Розробка фрагмента програми аудиту інформаційної безпеки	168
Контрольні питання до розділу 17	168

РОЗДІЛ 18. СТАНДАРТИ ТА МЕТОДИКИ АУДИТУ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ	169
18.1. ISO 19011: настанови щодо аудиту систем менеджменту	170
18.2. ISO/IEC 27001 і цикл сертифікаційного аудиту	170
18.3. NIST SP 800-53A: оцінювання заходів безпеки	171
18.4. Українська нормативна база аудиту інформаційної безпеки	171
18.5. Галузеві стандарти аудиту: PCI DSS та SOC 2	172
18.6. Чекліст-орієнтований проти ризик-орієнтованого аудиту	172
18.7. Матриця відповідності між стандартами (compliance mapping)	173
18.8. Вибір методики аудиту під конкретну задачу організації	173
18.9. Аудит третіх сторін і постачальників (third-party/vendor audit)	174
Практикум 18. Побудова матриці відповідності між ISO/IEC 27001 і НД ТЗІ	175
Контрольні питання до розділу 18	175

РОЗДІЛ 19. ЗАХОДИ ЩОДО АУДИТУ ЗАХИЩЕНИХ ІТ-СИСТЕМ: ЕТАПИ, ЗБІР ДОКАЗІВ, ЗВІТНІСТЬ	176
19.1. Життєвий цикл аудиторської перевірки	177

19.2. Планування аудиту й визначення області охоплення.....	178
19.3. Методи збору доказів аудиту	178
19.4. Робочі документи аудитора	179
19.5. Класифікація невідповідностей за критичністю.....	179
19.6. Складання звіту аудиту.....	180
19.7. План коригувальних дій і подальший контроль.....	180
19.8. Забезпечення якості аудиту	181
19.9. Комунікація результатів аудиту з керівництвом (closing meeting).....	181
Практикум 19. Проведення навчального аудиту фрагмента конфігурації безпеки	182
Контрольні питання до розділу 19.....	182

РОЗДІЛ 20. АУДИТ З ВИКОРИСТАННЯМ СКАНЕРІВ БЕЗПЕКИ (ПОГЛИБЛЕНО).....183

20.1. Роль автоматизованого сканування в програмі аудиту	184
20.2. Класи сканерів безпеки	184
20.3. Планування й авторизація сканування	185
20.4. Інтерпретація та підтвердження результатів сканування.....	185
20.5. Сканування вразливостей проти тестування на проникнення.....	186
20.6. Автоматизація регулярного сканування в програмі аудиту	186
20.7. Комплаєнс-сканування конфігурації	187
20.8. Обмеження автоматизованого сканування і роль аудитора.....	187
20.9. Сканування вебзастосунків (DAST) як спеціалізований клас сканерів	188
Практикум 20. Комплексне сканування навчального середовища для потреб аудиту	188
Контрольні питання до розділу 20.....	189

РОЗДІЛ 21. АУДИТ ЦІЛІСНОСТІ ФАЙЛОВИХ СИСТЕМ190

21.1. Поняття цілісності даних і файлової системи	190
21.2. Принцип роботи File Integrity Monitoring (FIM).....	191
21.3. Інструменти FIM.....	192
21.4. Побудова базового профілю файлової системи	192
21.5. FIM у вимогах стандартів аудиту	193
21.6. Хибнопозитивні спрацювання FIM і тюнінг	193
21.7. Цілісність у хмарному й контейнеризованому середовищі	194
21.8. Цілісність баз даних: контроль на рівні СУБД.....	195
Практикум 21. Розгортання File Integrity Monitoring для критичного каталогу	195
Контрольні питання до розділу 21	196

**РОЗДІЛ 22. ДОСЛІДЖЕННЯ ЗАСОБІВ ПЕРЕШКОДЖАННЯ АУДИТУ
ІНФОРМАЦІЇ В ІТ-СИСТЕМІ.....196**

22.1. Поняття протидії аудиту та мотивація зловмисника	197
22.2. Очищення й підробка журналів	198
22.3. Приховування даних: стеганографія, альтернативні потоки, timestomping	198
22.4. Ухилення від сканерів і засобів аудиту	199
22.5. Захисні заходи проти протидії аудиту.....	199
22.6. Виявлення ознак протидії аудиту під час перевірки.....	200
22.7. Юридичні наслідки навмисної протидії аудиту	200
22.8. Протидія аудиту в хмарному середовищі: видалення CloudTrail/Activity Logs.....	201

Практикум 22. Виявлення слідів протидії аудиту в навчальному середовищі	203
Контрольні питання до розділу 22	203
РОЗДІЛ 23. AI-АНАЛІЗ ПОДІЙ БЕЗПЕКИ: МАШИННЕ НАВЧАННЯ, UEBA В SIEM/SOC.....	204
23.1. Типи задач машинного навчання, застосовні до аналізу подій безпеки	205
23.2. UEBA: поглиблений механізм побудови базової поведінки	205
23.3. Конвеєр обробки подій: від сирих даних до пріоритизованого інциденту	206
23.4. Обмеження моделей: дрейф, змагальне машинне навчання, надмірна довіра	207
23.5. Генеративний ШІ та великі мовні моделі в SOC	208
23.6. Автономні AI-агенти для виявлення інцидентів	208
23.7. Оцінювання якості AI-аналізу: точність, повнота, F1	209
23.8. Етичні та регуляторні аспекти застосування ШІ в безпеці	209
23.9. SOAR і автоматизоване реагування на основі AI-пріоритизації	210
Практикум 23. Оцінювання якості моделі виявлення аномалій на навчальному наборі даних	212
Контрольні питання до розділу 23	212
РОЗДІЛ 24. КОМПЛЕКСНИЙ ПРАКТИКУМ: ПОБУДОВА SOC-ПРОЦЕСУ МОНІТОРИНГУ, АУДИТУ ТА РЕАГУВАННЯ.....	213
24.1. Карта підручника: як розділи 1-23 складаються в єдиний SOC-процес	214
24.2. Модель зрілості SOC	215
24.3. Проєктування SOC для гіпотетичної організації: вихідні умови	215
24.4. Вибір технологічного стеку та обґрунтування рішень	216
24.5. Робочий процес обробки інциденту від виявлення до закриття	216
24.6. Панель ключових показників ефективності SOC	217
24.7. Перспективи розвитку курсу та подальшого вдосконалення SOC	218
24.8. Юридична та етична відповідальність SOC-аналітика	218
Практикум 24. Комплексне проєктування SOC-процесу для гіпотетичної організації	220
Контрольні питання до розділу 24	220
ДОВІДКОВИЙ АПАРАТ	221
Глосарій термінів.....	221
Терміни Модуля 1 (моніторинг)	221
Терміни Модуля 2 (адміністрування)	222
Терміни Модуля 3 (аудит)	223
Перелік стандартів, згаданих у підручнику	226
Словник абревіатур.....	227
Розширений перелік контрольних питань до іспиту.....	228
Модуль 1. Моніторинг захищених ІТ-систем і мереж	228
Модуль 2. Адміністрування захищених ІТ-систем і мереж	234
Модуль 3. Аудит захищених ІТ-систем і мереж	241
Додаток. Приклади типових конфігурацій та запитів	249
Зведений список використаних джерел	252

МОДУЛЬ 1. МОНІТОРИНГ ЗАХИЩЕНИХ ІТ-СИСТЕМ І МЕРЕЖ

Відповідає Змістовому модулю 1 робочої програми: Тема 1 «Моніторинг захищених ІТ систем і мереж та системи безпеки», Тема 2 «Виявлення мережеских атак шляхом аналізу трафіку. Основи застосування DLP-систем».

РОЗДІЛ 1. ОСНОВИ МОНІТОРИНГУ ЗАХИЩЕНИХ ІТ-СИСТЕМ І МЕРЕЖ



Зміст розділу

- 1.1. Поняття, мета та завдання моніторингу інформаційної безпеки**
- 1.2. Об'єкти, суб'єкти та рівні моніторингу**
- 1.3. Нормативно-правові засади моніторингу інформаційної безпеки**
- 1.4. Метрики та показники ефективності моніторингу**
- 1.5. Моніторинг мережевого обладнання за протоколом SNMP**
- 1.6. Активна перевірка доступності служб: ICMP та синтетичний моніторинг**
- 1.7. Побудова базової лінії нормальної поведінки системи (baseline)**
- 1.8. Платформи мережевого моніторингу: огляд практичних інструментів**
- 1.9. Виклики масштабування системи моніторингу у великих мережах**
- 1.10. Моніторинг як послуга: власний SOC проти аутсорсингу (MSSP)**
- 1.11. Інтеграція моніторингу із системою обліку заявок (ITSM)**

Служба інформаційної безпеки дізнається про витік бази клієнтів не в момент атаки, а через тиждень, коли викрадені дані вже продають на тіньовому форумі. Формально усі засоби захисту стояли на місці: міжмережеский екран, антивірус, розмежування доступу. Не було тільки одного: людини або системи, яка помітила б

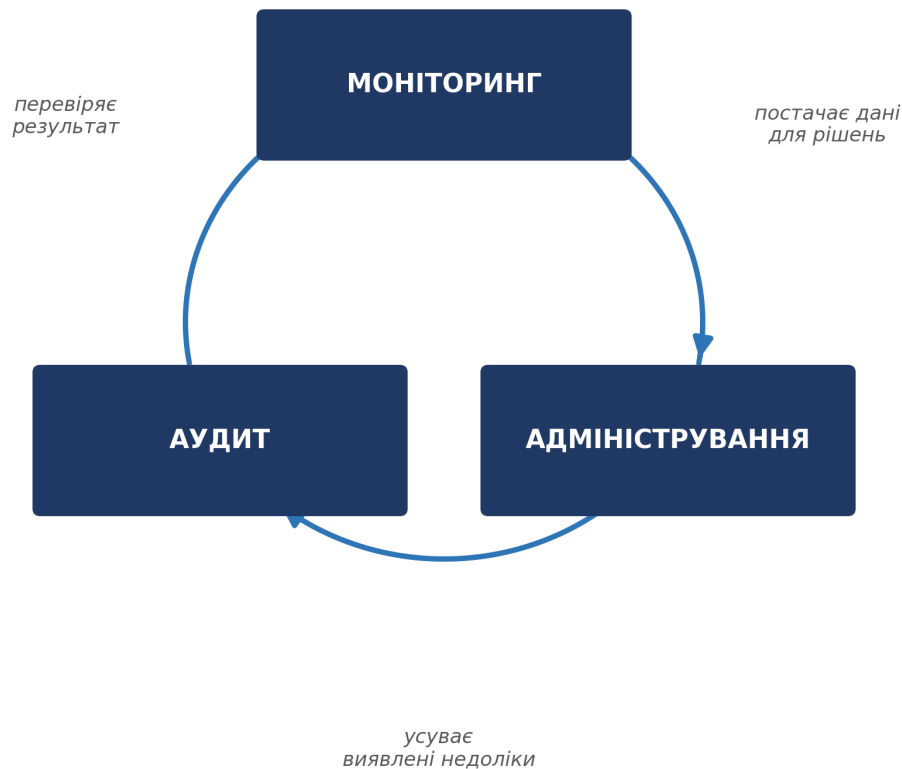
підозрілу активність, поки та ще відбувалася. Саме цю прогалину закриває моніторинг: без нього навіть найдорощчі засоби захисту працюють наосліп.

1.1. Поняття, мета та завдання моніторингу інформаційної безпеки

Моніторинг інформаційної безпеки — це безперервний або регулярний процес спостереження, збору, реєстрації та первинного аналізу даних про стан захищеності інформаційно-телекомунікаційної системи (ІТС) та про події, що в ній відбуваються. Мета процесу проста для формулювання і складна для реалізації: вчасно помітити відхилення від встановленої політики безпеки, вразливість або ознаку атаки, поки ще можна діяти, а не розслідувати наслідки.

Тут варто одразу розвести три поняття, які студенти й практики часто плутають. Моніторинг спостерігає за системою в реальному часі. Адміністрування змінює стан системи, впроваджуючи налаштування та засоби захисту. Аудит перевіряє відповідність системи вимогам на конкретний момент часу, вже після факту. Рисунок 1.1 показує, як ці три види діяльності утворюють єдиний, безперервний цикл: моніторинг постачає дані для рішень, адміністрування усуває виявлені недоліки, аудит перевіряє результат і ставить моніторингу нові завдання.

У міжнародній практиці ця функція має власне усталене позначення: **ISCM, Information Security Continuous Monitoring** (NIST SP 800-137). Стандарт визначає її як підтримання постійної обізнаності про інформаційну безпеку, вразливості та загрози, достатньої для прийняття управлінських рішень щодо ризиків. Формулювання варте уваги: мета моніторингу не «зібрати якомога більше логів», а забезпечити прийняття рішень. Система моніторингу, що генерує тисячі сповіщень на день, з якими ніхто не встигає працювати, не виконує свого призначення так само, як і система, що взагалі мовчить.



*Рис. 1.1. Місце моніторингу в циклі забезпечення кібербезпеки організації
Джерело: розроблено автором*

Розгорнемо мету моніторингу в перелік практичних завдань, які повинен уміти вирішувати майбутній фахівець з кібербезпеки:

1. безперервно спостерігати за функціонуванням ІТС і мережі, вчасно виявляючи відхилення від штатного режиму роботи;
2. виявляти спроби несанкціонованого доступу, мережеві атаки та інші порушення політики безпеки;
3. контролювати цілісність, конфіденційність і доступність інформаційних ресурсів;
4. знаходити вразливості компонентів ІТС раніше, ніж це зробить зловмисник;
5. накопичувати дані (події, журнали, метрики) для подальшого розслідування інцидентів та проведення аудиту;
6. формувати статистику й звітність для керівництва про фактичний стан захищеності.

1.2. Об'єкти, суб'єкти та рівні моніторингу

Питання «що саме моніторити» здається очевидним, поки не почнеш складати список. Сучасна ІТС складається з мережевого обладнання, серверів, робочих станцій, застосунків, баз даних і дедалі частіше хмарних сервісів, і кожен із цих рівнів генерує власний потік подій зі своєю логікою. Рисунок 1.2 узагальнює шість рівнів, на яких організовується моніторинг: від хмарної інфраструктури вгорі до дій конкретного користувача внизу.

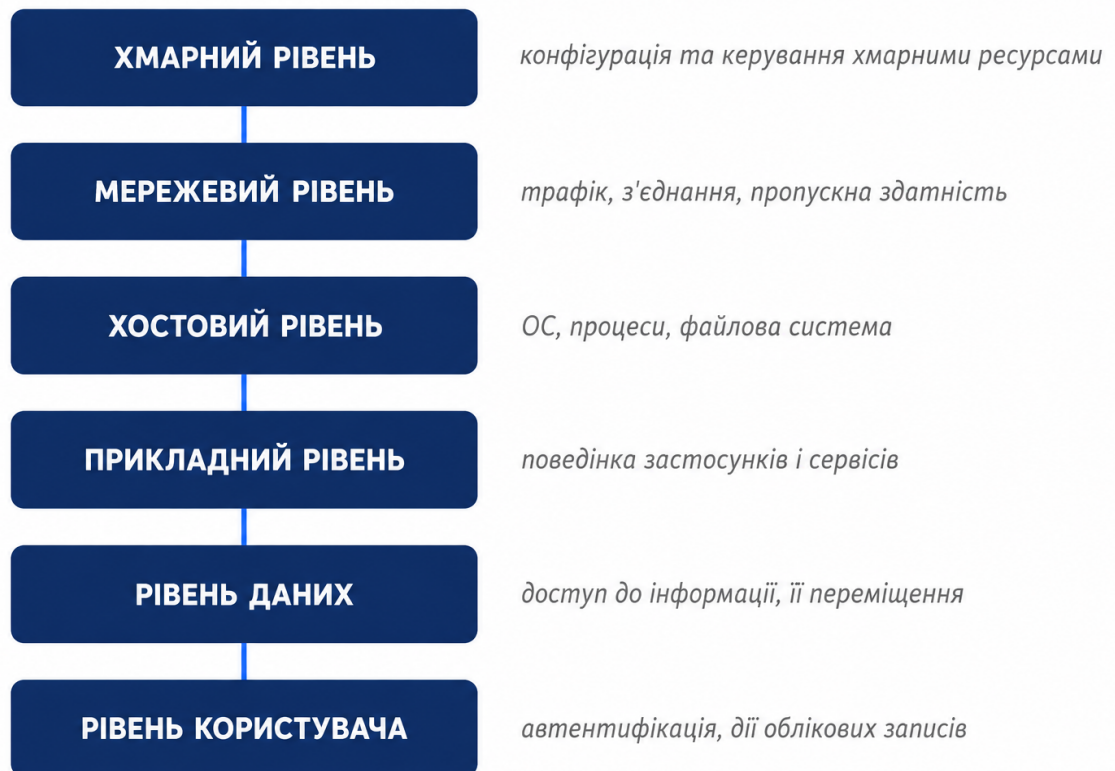


Рис. 1.2. Рівні моніторингу інформаційно-телекомунікаційної системи

Джерело: розроблено автором

Практичний висновок з рисунка 1.2 такий: SOC-аналітик, що дивиться лише на мережевий трафік, пропустить крадіжку даних через легітимний, але скомпрометований обліковий запис (рівень користувача), так само як адміністратор, що стежить тільки за обліковими записами, пропустить експлуатацію вразливості в застосунку (прикладний рівень). Повноцінна система моніторингу поєднує джерела з усіх рівнів одночасно, а не обирає один із них. Питання моніторингу телекомунікаційних мереж як окремого, суміжного з цією книгою напрямку, зокрема

на мережевому й каналному рівнях, детальніше розкрито в навчальному посібнику «Моніторинг телекомунікаційних мереж» [8] зі списку джерел цього розділу.

Суб'єктами моніторингу є люди й підрозділи, що виконують цю роботу: черговий адміністратор безпеки, SOC-аналітик рівня L1, L2 або L3, служба інформаційної безпеки організації в цілому. У великих організаціях моніторинг ведеться цілодобово, у режимі 24×7, силами власного Security Operations Center (SOC) або аутсорсингового постачальника послуг (MSSP, Managed Security Service Provider).

За способом отримання даних моніторинг поділяють на пасивний, коли спостереження не втручається в роботу системи (аналіз копії трафіку, читання журналів), та активний, коли система сама «опитує» об'єкти: сканує порти, надсилає SNMP-запити, перевіряє доступність служб. Другому підходу присвячено підрозділ 1.5.

1.3. Нормативно-правові засади моніторингу інформаційної безпеки

Моніторинг в українській організації не існує у вакуумі: він спирається на кілька рівнів регулювання. На законодавчому рівні це Закон України «Про захист інформації в інформаційно-телекомунікаційних системах» та Закон України «Про основні засади забезпечення кібербезпеки України» (2017), який визначає національну систему кібербезпеки та повноваження CERT-UA. На рівні технічного захисту інформації діють нормативні документи системи ТЗІ (НД ТЗІ). На рівні національних стандартів важливими є серія ДСТУ ISO/IEC 27000 та ДСТУ 3396.2-97. На міжнародному рівні орієнтирами лишаються ISO/IEC 27001, ISO/IEC 27002 та вже згаданий NIST SP 800-137.

Примітка. Нормативна база технічного захисту інформації в Україні періодично оновлюється, тому перед використанням конкретного документа варто перевірити його чинність за офіційними реєстрами Держспецзв'язку та національного органу стандартизації.

1.4. Метрики та показники ефективності моніторингу

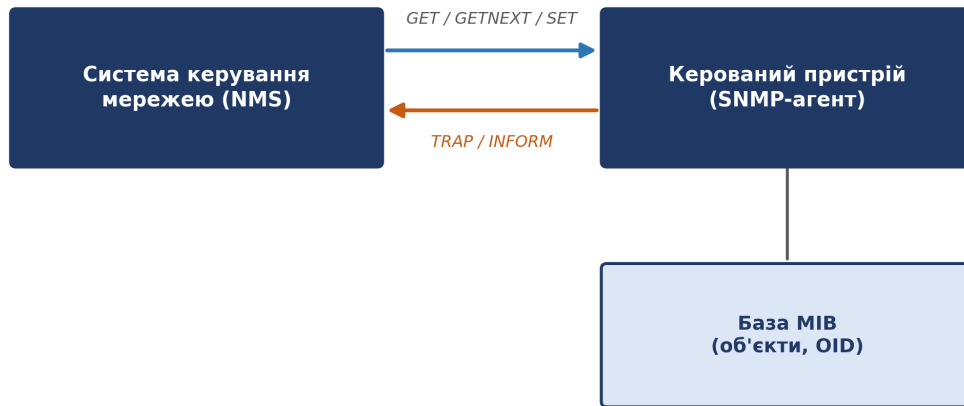
«У нас впроваджено моніторинг» саме по собі нічого не означає для керівника служби безпеки: важливо, наскільки він ефективний. Тому зрілість процесу оцінюють вимірюваними показниками, а не фактом наявності системи:

- **MTTD (Mean Time to Detect):** середній час від виникнення інциденту до його виявлення;
- **MTTR (Mean Time to Respond/Resolve):** середній час від виявлення до усунення наслідків;
- **частка хибнопозитивних спрацювань (False Positive Rate):** критично впливає на навантаження аналітиків і на «втому від сповіщень» (alert fatigue);
- **частка хибнонегативних спрацювань (False Negative Rate):** пропущені реальні інциденти, найдорожча з можливих помилок;
- **покриття джерел подій (coverage):** частка критичних систем, з яких події реально надходять до системи моніторингу;
- **інтенсивність потоку подій (EPS, Events Per Second):** базовий параметр для планування продуктивності майбутньої SIEM-системи (розділ 3).

Скорочення MTTD і MTTR є головним практичним критерієм зрілості SOC. Саме заради цього скорочення в наступних розділах книги буде послідовно нарощуватися інструментарій: від централізованого журналювання (розділ 2) до кореляції в SIEM (розділ 3) і врешті до методів штучного інтелекту (модуль 3). Питання автоматизованого скорочення MTTD і MTTR за рахунок ІІ-агентів, що аналізують потік телеметрії (SNMP, syslog, NetFlow) для виявлення інцидентів і помилок конфігурації без участі людини на первинному етапі, розглянуто в [22] зі списку джерел цього розділу.

1.5. Моніторинг мережевого обладнання за протоколом SNMP

Найстаріший і досі найпоширеніший спосіб активного моніторингу мережевого обладнання, тобто маршрутизаторів, комутаторів, принтерів, джерел безперебійного живлення, це протокол **SNMP (Simple Network Management Protocol)**. Його логіка зображена на рисунку 1.3: система керування мережею (NMS) опитує SNMP-агента на пристрої, а той відповідає значеннями з бази MIB (Management Information Base), ієрархічної структури об'єктів, де кожен має унікальний ідентифікатор OID.



SNMPv3 додає до цієї взаємодії USM-автентифікацію та шифрування трафіка керування

*Рис. 1.3. Взаємодія компонентів протоколу SNMP
Джерело: розроблено автором*

Основні операції протоколу: GET і GETNEXT запитують значення об'єкта, SET змінює значення, а TRAP/INFORM дозволяють агенту самому асинхронно повідомити NMS про подію, не чекаючи на запит. Версії протоколу суттєво різняться рівнем захищеності цієї взаємодії, що зведено в таблиці 1.1.

Архітектура управління послугами через протокол SNMP і місце цього протоколу в загальній системі управління мережею детально розглянуті в навчальному посібнику з грифом Міністерства освіти і науки України «Управління та якість послуг інформаційних мереж зв'язку», де окремий розділ присвячено саме прикладному рівню архітектури управління послугами на основі SNMP.

Оскільки community string у SNMPv1/v2c передається у відкритому вигляді, він так само вразливий до перехоплення, як пароль, відправлений без шифрування. У захищеній ІТС слід застосовувати виключно SNMPv3 або, якщо обладнання цього не підтримує, тунелювати трафік керування через захищений канал.

Таблиця 1.1. Порівняння версій протоколу SNMP

Версія	Автентифікація	Шифрування трафіка	Практична придатність
SNMPv1	Community string (відкритий текст)	Відсутнє	Застаріла, використовувати не рекомендовано
SNMPv2 с	Community string (відкритий текст)	Відсутнє	Прийнятна лише в ізольованих керуючих мережах
SNMPv3	USM: HMAC-MD5 або HMAC-SHA	DES або AES (за налаштування)	Єдиний варіант, придатний для захисної ІТС

Окремим і активно досліджуваним напрямом є моніторинг програмно-конфігурованих мереж (SDN), де класична модель SNMP-опитування доповнюється централізованим контролером, що безпосередньо бачить стан усієї мережі. Питання безпеки та діагностики несправностей при розгортанні SDN, а також аналізу програмного забезпечення для контролю за пакетами даних у таких мережах, детальніше розкрито в [4] та [5] зі списку джерел цього розділу. Архітектурні наслідки централізації керування мережею докладніше розглянуто в розділі 13 модуля 2.

1.6. Активна перевірка доступності служб: ICMP та синтетичний моніторинг

Поряд із SNMP-опитуванням, розглянутим у підрозділі 1.5, найпростішою й найстарішою формою активного моніторингу є перевірка доступності вузла за протоколом **ICMP (Internet Control Message Protocol)**, відома як ping: система моніторингу періодично надсилає ICMP Echo Request на контрольований вузол і за наявності та часом надходження відповіді Echo Reply робить висновок про доступність вузла й приблизну затримку мережі. Утиліта traceroute, що спирається на той самий протокол, додатково дозволяє визначити маршрут проходження пакета через проміжні вузли, що корисно для локалізації місця мережевої проблеми.

Обмеженість ICMP-моніторингу полягає в тому, що доступність вузла за мережею ще не означає працездатності конкретної служби, яка на ньому працює: сервер може відповідати на ping, тоді як застосунок, розгорнутий на ньому, вже завис. Тому сучасні системи моніторингу доповнюють ICMP-перевірку **перевіркою служб (service check)** на прикладному рівні: TCP-з'єднанням до конкретного порту,

HTTP-запитом до вебзастосунку з перевіркою коду відповіді, або повноцінною **синтетичною транзакцією (synthetic monitoring)**, коли система моніторингу емулює дії реального користувача (наприклад, послідовність «відкрити сторінку входу, авторизуватися, виконати пошук») і вимірює час виконання кожного кроку. Синтетичний моніторинг цінний саме тим, що виявляє деградацію сервісу ще до того, як на неї поскаржиться реальний користувач, і є природним доповненням до пасивного журналювання (розділ 2), яке фіксує лише те, що вже сталося.

1.7. Побудова базової лінії нормальної поведінки системи (baseline)

Більшість методів виявлення відхилень, від найпростішого порогового сповіщення до складних алгоритмів виявлення аномалій (розділ 6, розділ 23 модуля 3), спираються на попередньо визначену **базову лінію (baseline)**: кількісний опис того, як система поводить себе в нормальному, штатному режимі роботи. Без базової лінії система моніторингу не має критерію, за яким відрізнити нормальне навантаження від ознаки проблеми: завантаження процесора 80% може бути тривожним сигналом для сервера бази даних, що зазвичай працює на 20%, і водночас цілком штатним показником для обчислювального вузла, спеціально призначеного для пікових навантажень.

Побудова базової лінії типово передбачає збір метрик протягом репрезентативного періоду (як мінімум кількох тижнів, щоб врахувати денні й тижневі цикли активності), визначення статистичних характеристик нормальної поведінки (середнє значення, стандартне відхилення, характерні піки, наприклад щомісячне закриття звітного періоду) і подальше динамічне порівняння поточних показників із цією базовою лінією, а не з жорстко заданим фіксованим порогом. Практична складність полягає в тому, що базова лінія сама змінюється з часом (сезонність, зростання кількості користувачів, поступова зміна архітектури системи), тому її потрібно періодично переобчислювати, а не фіксувати одноразово під час впровадження системи моніторингу.

1.8. Платформи мережевого моніторингу: огляд практичних інструментів

Підрозділи 1.5-1.6 розглянули протоколи, на яких будується активний моніторинг; на практиці адміністратор рідко працює з SNMP чи ICMP напряду, а користується платформою мережевого моніторингу (Network Monitoring System, NMS), що інкапсулює ці протоколи у зручний інтерфейс зі сповіщеннями, графіками й історією. Серед поширених рішень з відкритим кодом виокремлюють **Zabbix**

(універсальна платформа з власним агентом і гнучкою системою тригерів), **Nagios** і його активно розвинений форк **Icinga** (класичні рішення, що історично зосереджені на бінарній перевірці «служба працює / не працює», доповнений плагінами), **LibreNMS** і **Cacti** (спеціалізовані передусім на SNMP-моніторингу мережевого обладнання з автоматичним виявленням топології). Комерційні рішення (наприклад PRTG Network Monitor) пропонують подібний функціонал з готовою підтримкою постачальника, що для багатьох організацій переважає вартість ліцензії.

Вибір між ними на практиці визначається не лише вартістю, а й моделлю збору даних: **агентний** підхід (окрема програма встановлюється на кожному контрольованому вузлі й активно надсилає дані на сервер) дає глибшу видимість усередині операційної системи, тоді як **безагентний** підхід (сервер моніторингу сам опитує вузол ззовні, за SNMP чи через WMI/SSH) простіший у розгортанні на пристроях, де встановлення стороннього програмного забезпечення небажане чи неможливе, наприклад на мережевому обладнанні чи спеціалізованих технологічних системах.

1.9. Виклики масштабування системи моніторингу у великих мережах

Система моніторингу, спроектована для десятків вузлів, рідко переноситься без змін на мережу з тисячами вузлів. Першим викликом є **інтервал опитування (polling interval)**: надто частий опит (наприклад, кожні 5 секунд для тисяч пристроїв) створює власне помітне навантаження на мережу й на сам сервер моніторингу, тоді як надто рідкий (раз на годину) робить систему нечутливою до короткочасних, але важливих подій. Практика полягає в диференціації інтервалу за критичністю об'єкта: критичні магістральні канали опитуються частіше, ніж периферійні пристрої другогорядного значення.

Другим викликом є топологія самого збору даних: у територіально розподіленій мережі централізований опитувальний сервер, розташований у головному офісі, змушений опитувати віддалені майданчики через канали зв'язку, які самі можуть бути нестабільними, що спотворює результат (недоступність через збій каналу моніторингу невідрізняє від недоступності самого вузла без додаткового аналізу). Розв'язком є **розподілена архітектура збору**: локальні збирачі (collectors, пробники) на кожному майданчику виконують опитування локально й передають на центральний сервер лише агреговані чи попередньо оброблені дані. Третім, часто недооціненим викликом є захищеність і відмовостійкість самої системи моніторингу: сервер, що бачить стан усієї інфраструктури, є привабливою

ціллю для зловмисника, а його власний відказ («хто спостерігає за спостерігачем») залишає організацію сліпою саме в момент, коли видимість найпотрібніша, тому критичні системи моніторингу самі потребують резервування, розглянутого в контексті RPO/RTO в підрозділі 11.5 модуля 2.

1.10. Моніторинг як послуга: власний SOC проти аутсорсингу (MSSP)

Підрозділ 1.2 згадав MSSP (Managed Security Service Provider) як альтернативу власному SOC; розглянемо цей вибір детальніше, оскільки він визначає всю подальшу організацію функції моніторингу. Побудова власного SOC 24×7 вимагає щонайменше кількох змінних аналітиків, інвестицій у платформу SIEM (розділ 3) і постійного навчання персоналу під нові загрози, що для середньої за розміром організації часто економічно невиправдано відносно обсягу подій, який реально потребує цілодобової уваги. **MSSP** пропонує готову інфраструктуру моніторингу й команду аналітиків на аутсорсинговій основі, зазвичай за суттєво нижчою сукупною вартістю, ніж власний SOC порівнюваної якості.

Зворотним боком аутсорсингу є втрата частини контролю й глибокого контексту: зовнішній аналітик MSSP не знає внутрішньої специфіки бізнес-процесів організації так, як власний співробітник, і при виявленні інциденту потребує додаткового часу на з'ясування контексту, що може вплинути на показник MTTR (підрозділ 1.4). Поширеним компромісом на практиці є **гібридна модель**: базовий моніторинг і первинна фільтрація подій (рівень L1) виконуються MSSP, тоді як поглиблене розслідування підтверджених інцидентів (L2/L3) і стратегічні рішення лишаються за невеликою власною командою організації, яка й формулює вимоги до провайдера.

1.11. Інтеграція моніторингу із системою обліку заявок (ITSM)

Виявлений засобами моніторингу інцидент чи аномалія рідко залишається виключно в межах SIEM-платформи: для реального усунення її потрібно передати відповідальному співробітнику, відстежити хід виконання і зафіксувати результат, а саме цю функцію в більшості організацій виконує окрема **система обліку заявок (ITSM, IT Service Management)**, наприклад ServiceNow чи Jira Service Management. Пряма інтеграція SIEM з ITSM-платформою автоматично створює заявку (тікет) одразу після спрацювання аналітичного правила, з попередньо заповненими полями (пріоритет, залучена система, стислий опис), що усуває затримку, характерну для ручного повідомлення адміністратора електронною поштою чи месенджером.

Така інтеграція також розв’язує практичну проблему подвійного обліку: без єдиної точки реєстрації заявок один і той самий інцидент може одночасно фіксуватися в кількох незалежних чергах різних команд, що ускладнює підрахунок реальних показників MTTD/MTTR (підрозділ 1.4) і згодом заплує аудит реагування на інцидент (підрозділ 19.3 модуля 3). Життєвий цикл заявки в ITSM, від створення через призначення відповідального до закриття з підтвердженням рішенням, безпосередньо повторює послідовність кроків процесу управління змінами, розглянутого в підрозділі 9.5 модуля 2, що робить ITSM природною сполучною ланкою між моніторингом (модуль 1) і адмініструванням (модуль 2) як практиками, а не лише теоретичними складовими одного циклу.

Практикум 1. Розгортання сканера вразливостей і базове сканування мережевого сегмента

Активний моніторинг не обмежується протоколом SNMP: окрему й дуже практичну групу інструментів становлять сканери безпеки, які автоматично перевіряють вузли мережі на наявність вразливостей і невідповідностей політиці безпеки. Розрізняють мережеве сканування (виявлення активних вузлів і відкритих портів, наприклад засобами Nmap), сканування вразливостей (зіставлення версій ПЗ з базою CVE, наприклад Nessus або OpenVAS/Greenbone) і сканування відповідності еталонним профілям (CIS Benchmarks, DISA STIG).

Мета роботи: отримати практичні навички розгортання сканера вразливостей і первинної інтерпретації результатів сканування.

Хід роботи:

1. розгорнути сканер (наприклад, OpenVAS/Greenbone Community Edition або Nessus Essentials) у тестовому сегменті мережі, ізольованому від продуктивних систем;
2. визначити діапазон адрес і політику сканування, для навчальних цілей обравши повне (deep) сканування;
3. виконати сканування контрольної групи з двох-трьох віртуальних машин із навмисно застарілим програмним забезпеченням;
4. класифікувати виявлені вразливості за рівнем критичності CVSS, окремо відзначивши ймовірні хибнопозитивні спрацювання;
5. підготувати короткий звіт із рекомендаціями щодо усунення трьох найкритичніших вразливостей.

Примітка для викладача. Конкретні кроки встановлення й ліцензійні умови продукту потрібно звіряти з чинною документацією виробника на момент проведення заняття.

Контрольні питання до розділу 1

1. Чим моніторинг принципово відрізняється від аудиту та адміністрування?
2. Які рівні (об'єкти) моніторингу ІТС зображено на рисунку 1.2 і чому жоден з них не можна ігнорувати?
3. Що таке ISCM і яким документом NIST визначено це поняття?

4. Назвіть основні нормативні рівні регулювання моніторингу інформаційної безпеки в Україні.
5. Що таке MTDD і MTTR і чому саме ці показники обрані як головний критерій зрілості SOC?
6. Які операції передбачає протокол SNMP? Чим SNMPv3 відрізняється від SNMPv1/v2c і чому це критично для захищеної ІТС?
7. Що таке SOC і чим відрізняється власний SOC від MSSP?
8. Поясніть різницю між пасивним і активним моніторингом і наведіть приклад кожного.
9. Що таке alert fatigue і чому система, яка генерує занадто багато сповіщень, не виконує свого призначення?
10. Дайте означення показників False Positive Rate і False Negative Rate. Який з них зазвичай «дорожчий» для організації і чому?
11. Що таке EPS (Events Per Second) і навіщо цей параметр потрібен при плануванні SIEM-системи?
12. Порівняйте версії протоколу SNMP за таблицею 1.1: чому SNMPv1/v2c не рекомендовано використовувати в захищеній ІТС?
13. Які операції SNMP дозволяють пристрою самостійно, асинхронно повідомити систему керування про подію?
14. Назвіть три причини, чому організація впроваджує моніторинг ІТС.
15. На яких нормативних рівнях (від закону до міжнародного стандарту) регулюється моніторинг інформаційної безпеки в Україні?
16. Опишіть практичний сценарій, коли SOC-аналітик, який стежить лише за мережевим трафіком, пропустить реальний інцидент.
17. Які типи сканування безпеки ви можете назвати? Наведіть приклади інструментів для кожного типу.

РОЗДІЛ 2. ЖУРНАЛЮВАННЯ ТА ЦЕНТРАЛІЗОВАНИЙ ЗБІР ПОДІЙ БЕЗПЕКИ



Розділ 2 · Журналювання та централізований збір подій

Зміст розділу

- 2.1. Роль журналювання в забезпеченні кібербезпеки***
- 2.2. Джерела подій безпеки***
- 2.3. Формати та протоколи журналювання***
- 2.4. Архітектура конвеєра централізованого збору подій***
- 2.5. Нормалізація та збагачення подій***
- 2.6. Політики зберігання та забезпечення цілісності журналів***
- 2.7. Журналювання в хмарному та контейнеризованому середовищі***
- 2.8. Управління обсягом журналів: багаторівневе зберігання та фільтрація малоцінних подій***
- 2.9. Поглиблене журналювання кінцевих точок: за межами стандартного журналу ОС***
- 2.10. Журнал як судовий доказ: цілісність ланцюга зберігання***
- 2.11. Стиснення та архівування історичних журналів***

Журнали подій нерідко називають «чорною скринькою» інформаційної системи за аналогією з бортовим самописцем літака: у штатному режимі на них ніхто не звертає уваги, але саме вони стають головним, а часто єдиним джерелом істини, коли щось пішло не так. Слідчий з кіберінцидентів, що приходить розслідувати злом, ставить одне й те саме перше запитання: «які журнали велися і скільки вони зберігаються?» Якщо відповідь незадовільна, розслідування може зупинитися вже на цьому кроці.

2.1. Роль журналювання в забезпеченні кібербезпеки

Журналювання (логування) — це фіксація подій, що відбуваються в компонентах ІТС, у вигляді структурованих або напівструктурованих записів. Керівний документ NIST SP 800-92 (*Guide to Computer Security Log Management*) визначає керування журналами як процес, що охоплює генерацію, передавання,

зберігання, аналіз і видалення даних аудиту безпеки. З цього визначення випливають чотири практичні завдання, які журналювання вирішує одночасно:

1. **виявлення інцидентів:** журнали є первинним джерелом подій для SIEM-кореляції (розділ 3);
2. **розслідування (forensics):** відновлення точної хронології дій зловмисника;
3. **підзвітність (accountability):** прив'язка кожної дії до конкретного облікового запису;
4. **відповідність вимогам (compliance):** стандарти на кшталт ISO/IEC 27001 (додаток А.8.15) чи PCI DSS прямо вимагають вести й зберігати журнали визначений термін.

2.2. Джерела подій безпеки

Джерела подій так само різноманітні, як і рівні моніторингу з розділу 1. Таблиця 2.1 систематизує основні категорії джерел разом із типовими прикладами подій, які вони постачають.

Таблиця 2.1. Основні джерела подій безпеки в типовій організації

Джерело	Приклади подій
Операційні системи	вхід/вихід користувача, зміна привілеїв, запуск процесів (Windows Event Log, Linux syslog/auditd)
Мережеве обладнання	дозволи/блокування трафіку міжмережовим екраном, зміни конфігурації маршрутизатора
Застосунки та СУБД	помилки автентифікації, запити до бази даних, зміни схеми
Засоби захисту	спрацювання IDS/IPS, антивірусу, DLP
Хмарні платформи	зміни конфігурації ресурсів, дії в панелі керування (control plane), доступ до сховищ
Ідентифікаційна інфраструктура	Active Directory / Entra ID: автентифікація, видача токенів, зміни груп і ролей

2.3. Формати та протоколи журналювання

Розмаїття джерел породжує розмаїття форматів, і без розуміння цих форматів наступний розділ про SIEM залишиться абстракцією. Головні з них: **Syslog** (RFC 5424), історично найпоширеніший протокол передавання подій з мережевого обладнання й Unix-подібних систем, з рівнями критичності (severity 0-7) і

категоріями джерела (facility); **Windows Event Log** разом із механізмом **Windows Event Forwarding**, власний XML-формат подій Windows із централізованим пересиланням без встановлення агента на кожному вузлі; **CEF (Common Event Format)**, розроблений компанією ArcSight формат нормалізації подій для SIEM; **LEEF (Log Event Extended Format)**, аналогічний формат, що застосовується в IBM QRadar; і, нарешті, **JSON**, який де-факто став стандартом для сучасних хмарних сервісів і мікросервісних застосунків.

2.4. Архітектура конвеєра централізованого збору подій

Один сервер зі своїм локальним журналом ще можна проглянути вручну. Тисяча серверів, комутаторів і хмарних сервісів вже ні: потрібен конвеєр, що автоматично збирає, обробляє і зберігає події з усіх джерел одночасно. Типова архітектура такого конвеєра показана на рисунку 2.1.



*Рис. 2.1. Конвеєр централізованого збору подій безпеки
Джерело: розроблено автором*

Кожна ланка конвеєра на рисунку 2.1 виконує окрему роль. Джерело, це host, мережевий пристрій, застосунок або хмарний сервіс, генерує подію. Агент, або forwarder (Filebeat, Winlogbeat, Fluent Bit, Fluentd, Splunk Universal Forwarder), зчитує локальні журнали й передає їх далі. Брокер повідомлень, наприклад Apache Kafka, застосовується у великих інфраструктурах як буфер, що запобігає втраті подій при піках навантаження, і є не обов'язковою ланкою для невеликих організацій. Обробник (Logstash, Fluentd) виконує нормалізацію, збагачення й фільтрацію. Сховище подій, індексована база даних на кшталт Elasticsearch, Splunk-індексів чи Log Analytics Workspace, оптимізоване під запис і повнотекстовий пошук. І нарешті рівень аналізу, SIEM-система, про яку піде мова в розділі 3.

2.5. Нормалізація та збагачення подій

Сирі події з різних джерел мають різні поля, різні назви для однакових понять і навіть різні формати часу: без приведення їх до спільного знаменника коректна кореляція неможлива. Рисунок 2.2 ілюструє цю проблему на прикладі трьох подій автентифікації з трьох різних джерел, кожна зі своїм форматом, які **нормалізація** зводить до єдиної схеми полів.

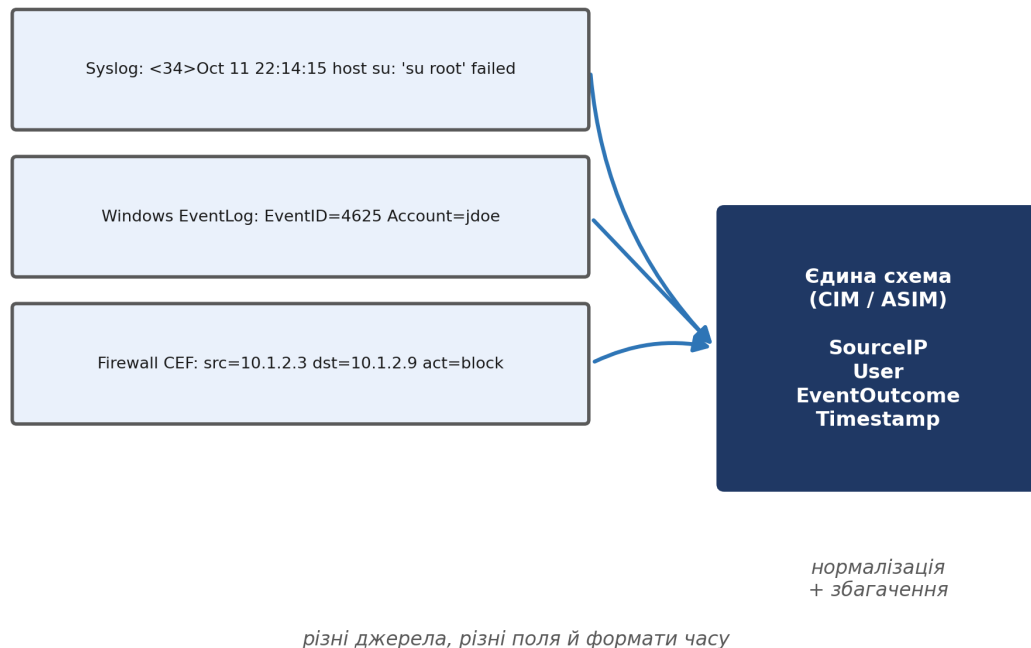


Рис. 2.2. Нормалізація подій із різних джерел до єдиної схеми полів
Джерело: розроблено автором

У Splunk цю роль виконує **CIM (Common Information Model)**, у Microsoft Sentinel, **ASIM (Advanced Security Information Model)**: обидва підходи детально розглянуто в розділі 3. Окремим кроком є **збагачення (enrichment)**, тобто доповнення події контекстною інформацією, якої немає в самому журналі: геолокацією IP-адреси, репутацією домену, зіставленням з даними Threat Intelligence (розділ 8) або приналежністю користувача до підрозділу.

2.6. Політики зберігання та забезпечення цілісності журналів

Централізований збір подій втрачає сенс, якщо журнали можна непомітно підчистити або якщо вони зникають раніше, ніж інцидент буде виявлено. Тому до

зберігання журналів безпеки висувають окремі вимоги. Термін зберігання визначається політикою організації та вимогами регуляторів: для критичної інфраструктури зазвичай не менше шести-дванадцяти місяців у «гарячому» доступі та довше в архіві. Захист від модифікації забезпечують моделлю WORM (Write Once Read Many) або регулярним хешуванням журналів із фіксацією часу, щоб унеможливити приховування слідів атаки через редагування логів. Синхронізація часу за єдиним джерелом (NTP) є обов'язковою умовою: без неї кореляція подій за часовою міткою стає ненадійною. Розмежування доступу передбачає, що журнали безпеки зберігаються окремо від систем, які їх генерують, і недоступні для адміністраторів цих систем, це і є принцип розділення обов'язків.

2.7. Журналювання в хмарному та контейнеризованому середовищі

Перенесення інфраструктури в хмару (детально розглянуто в розділі 14 модуля 2) суттєво змінює саму природу журналювання: замість фіксованого набору серверів з операційною системою, що працює місяцями, адміністратор має справу з ефемерними ресурсами, контейнерами (розділ 15 модуля 2) і віртуальними машинами, що можуть існувати лічені хвилини. Хмарні платформи надають власні, специфічні служби журналювання: наприклад, журнал звернень до API керування хмарою (у AWS - CloudTrail, у Azure - Activity Log), який фіксує кожну дію над хмарними ресурсами (створення, зміну, видалення) незалежно від того, хто і як цю дію виконав, через консоль, командний рядок чи автоматизований скрипт Infrastructure as Code.

Для контейнеризованих застосунків (Kubernetes) традиційний підхід «журнал у файл на диску вузла» не працює: под може бути видалений і перепланований на інший вузол за лічені секунди, а разом з ним зникає і локальний журнал. Тому стандартною практикою є передавання виводу контейнера (stdout/stderr) безпосередньо в конвеєр централізованого збору подій, розглянутий у підрозділі 2.4, часто через проміжний агент, розгорнутий на кожному вузлі кластера (DaemonSet у термінології Kubernetes). Окремо варто журналювати **audit log Kubernetes**, службу, що фіксує кожен запит до API-сервера кластера (розділ 15 модуля 2), аналогічно до журналу дій адміністратора в традиційній ІТС.

2.8. Управління обсягом журналів: багаторівневе зберігання та фільтрація малоцінних подій

Зі зростанням кількості джерел (підрозділ 2.2) обсяг журналів, що надходять до конвеєра, зростає непропорційно швидко: сучасна SIEM-система середньої організації легко приймає десятки чи сотні гігабайтів подій на добу. Оскільки більшість SIEM-платформ тарифікуються за обсяг проковтнутих (ingest) або проіндексованих даних (розділи 3 і 5), некерований потік подій перетворюється на пряму й швидко зростаючу статтю витрат, а не лише на технічну задачу.

Практичне управління обсягом спирається на кілька взаємодоповнюючих підходів. **Багаторівневе зберігання (tiering)** розділяє дані за віком і цінністю: свіжі події останніх днів чи тижнів лишаються в «гарячому» шарі з повнотекстовим індексуванням і швидким пошуком, старіші події переносяться в дешевший «холодний» шар (архівне сховище) з повільнішим, але значно дешевшим доступом, необхідним переважно для розслідування давніх інцидентів чи виконання вимог зберігання з підрозділу 2.6. **Фільтрація на джерелі (filtering at the edge)** відсіює завідомо малоцінні події (наприклад, службові діагностичні повідомлення рівня debug) ще до передавання в конвеєр, замість того щоб платити за їхнє індексування і потім ігнорувати під час пошуку. **Сумаризація (summarization)**, коли замість кожної окремої однотипної події зберігається агрегований підрахунок за проміжок часу, застосовна до подій з дуже високою частотою й низькою індивідуальною цінністю. Головний ризик надмірної фільтрації, натомість, полягає в тому, що подія, яка сьогодні здається неважливою, може виявитися критичним доказом під час розслідування інциденту, що станеться через кілька місяців, тому рішення про виключення джерела з потоку журналювання повинне прийматися свідомо, а не як побічний ефект економії.

2.9. Поглиблене журналювання кінцевих точок: за межами стандартного журналу ОС

Стандартний журнал подій операційної системи (Windows Event Log чи syslog, згадані в підрозділі 2.3) реєструє передбачений виробником перелік подій, здебільшого пов'язаних із входом користувача й роботою самої системи, але лишає поза увагою значну частину активності, яка найбільше цікавить аналітика при полюванні на загрози (підрозділ 8.8): запуск конкретного процесу з повним переліком аргументів командного рядка, мережеві з'єднання, ініційовані процесом, зміни в реєстрі. Для Windows цю прогалину закриває безкоштовний системний засіб

Sysmon (System Monitor) із набору Sysinternals: після встановлення й налаштування за конфігураційним XML-файлом Sysmon генерує суттєво деталізованіші події (створення процесу з хешем виконуваного файлу, мережеве з'єднання, зміну часової мітки файлу) безпосередньо в журнал подій Windows, звідки вони підхоплюються тим самим конвеєром збору, розглянутим у підрозділі 2.4.

Для Linux аналогічну роль виконує підсистема **auditd**, вбудована в ядро, що за визначеними адміністратором правилами аудиту фіксує системні виклики: відкриття файлу, зміну прав доступу, виконання команди від імені root. Практична цінність такого поглибленого журналювання для книги в цілому висока: саме дані Sysmon чи auditd, а не стандартний журнал ОС, найчастіше лежать в основі правил кореляції SIEM (розділ 3) і сигнатур виявлення на основі MITRE ATT&CK (підрозділ 8.4), оскільки саме вони фіксують технічні деталі дії злоумисника, а не лише факт його входу в систему.

2.10. Журнал як судовий доказ: цілісність ланцюга зберігання

Підрозділ 2.6 розглянув технічний захист журналів від модифікації (WORM, хешування); розглянемо тепер, чому це не суто технічна вимога, а передумова придатності журналу як доказу в розслідуванні, зокрема в разі, якщо інцидент переростає в судову справу. Поняття **ланцюга зберігання доказів (chain of custody)**, запозичене з класичної криміналістики, вимагає документально підтвердженої, безперервної послідовності: хто, коли й яким чином отримав доступ до журналу, чи вносились у нього будь-які зміни, де й як він зберігався від моменту фіксації події до моменту подання як доказу. Розрив у цьому ланцюзі, навіть суто формальний, дає стороні захисту підставу поставити під сумнів достовірність доказу цілком, незалежно від того, наскільки переконливим виглядає сам зміст журналу.

Практичним наслідком для адміністратора є принцип **готовності до судового розгляду (forensic readiness)**: організація, яка чекає на настання інциденту, щоб почати думати про належне документування журналів, як правило, вже запізнюється. Готовність передбачає заздалегідь визначену процедуру вилучення й підпису (наприклад, криптографічного хешування) копії журналу в момент виявлення інциденту, призначену відповідальну особу за збереження цієї копії окремо від оперативної системи журналювання, і документування кожної дії з доказом з фіксацією часу й виконавця, тобто застосування того самого принципу підзвітності, що вводився в підрозділі 2.1, але вже до самого процесу розслідування, а не лише до вихідної події.

2.11. Стиснення та архівування історичних журналів

Політика зберігання, розглянута в підрозділі 2.6, визначає, як довго журнал повинен лишатися доступним, але не вирішує окремого практичного питання: більшість подій, старших за кілька тижнів, майже ніколи не потрапляють у щоденні запити аналітика, тому зберігання їх у «гарячому», повністю індексованому стані в SIEM (розділ 3) не виправдано витрачає найдорожчий ресурс платформи. Типове рішення, багаторівневе архівування, переносить дані за визначеним графіком з гарячого сховища в дешевше, стиснене «холодне» (наприклад, об'єктне хмарне сховище чи стрічковий носій), зберігаючи водночас можливість повторного імпорту в разі потреби, наприклад під час позапланового розслідування чи планового аудиту (розділ 19 модуля 3).

Ключовий компроміс цього підходу: час отримання даних із холодного архіву (rehydration time) вимірюється годинами, а не секундами, тому графік архівування повинен враховувати не лише вимоги регуляторів до мінімального терміну зберігання (підрозділ 1.3), а й реалістичну оцінку того, наскільки швидко організації може знадобитися доступ до конкретної категорії старих подій. Стиснення без втрат (lossless), застосовуване практично завжди для журналів, зазвичай скорочує обсяг у 5-10 разів завдяки високій повторюваності текстових структур типового журналу події, що суттєво знижує вартість довгострокового зберігання порівняно з наївним підходом «зберігати все і завжди в гарячому вигляді».

Практикум 2. Побудова конвеєра централізованого збору подій

Мета роботи: налаштувати пересилання подій із вузла-джерела до центрального колектора та переконатися в коректності нормалізації.

Хід роботи:

1. налаштувати на тестовому Linux-вузлі пересилання syslog на колектор (rsyslog або syslog-ng у пару з Logstash чи Fluent Bit);
2. на тестовому Windows-вузлі налаштувати Windows Event Forwarding або встановити легкий агент (Winlogbeat);
3. перевірити надходження подій у сховище (Elasticsearch або Log Analytics) і коректність часових міток у форматі UTC;
4. написати простий фільтр нормалізації, що виокремлює поля `src_ip`, `user`, `event_id`, і перевірити його на потоці реальних подій;
5. оформити короткий звіт: які поля вдалося нормалізувати автоматично, а які знадобилося парсити вручну.

Контрольні питання до розділу 2

1. Чому журнали вважаються первинним доказовим матеріалом під час розслідування інцидентів?
2. Перелічіть основні джерела подій безпеки в типовій організації (таблиця 2.1).
3. Чим формат CEF відрізняється від «сирого» повідомлення syslog?
4. Які шість ланок утворюють конвеєр централізованого журналювання, зображений на рисунку 2.1?
5. Що таке нормалізація подій і чому без неї неможлива коректна кореляція? Наведіть приклади (CIM, ASIM).
6. Які вимоги висуваються до забезпечення цілісності журналів безпеки?
7. Назвіть щонайменше чотири типові джерела подій безпеки в організації.
8. Чим структурований формат журналювання (наприклад, JSON) відрізняється від неструктурованого текстового повідомлення?
9. Яку роль виконує агент або forwarder у конвеєрі централізованого збору подій (рисунок 2.1)?
10. Навіщо потрібен проміжний брокер повідомлень у деяких архітектурах збору журналів?
11. Що таке часова синхронізація (NTP) і чому вона критична для кореляції подій з різних джерел?

12. Наведіть приклад єдиної схеми нормалізації (СІМ або АСІМ) і поясніть, які поля вона визначає.
13. Скільки часу типово потрібно зберігати журнали безпеки і від чого залежить цей термін?
14. Що означає забезпечення цілісності журналів і якими технічними засобами воно досягається?
15. Чому «сирий» (raw) журнал варто зберігати навіть після нормалізації?
16. Опишіть різницю між журналюванням методом push (агент сам надсилає дані) і pull (система опитує джерело).

РОЗДІЛ 3. SIEM-СИСТЕМИ: АРХІТЕКТУРА, ФУНКЦІЇ, ОГЛЯД РИНКУ РІШЕНЬ



Розділ 3 · SIEM-системи: архітектура, функції, ринок рішень

Зміст розділу

- 3.1. Визначення SIEM та еволюція технології**
- 3.2. Архітектура SIEM**
- 3.3. Методи кореляції подій**
- 3.4. Огляд ринку рішень**
- 3.5. Критерії вибору SIEM для організації**
- 3.6. Sigma: універсальна мова опису правил виявлення**
- 3.7. UEBA: аналітика поведінки користувачів і сутностей**
- 3.8. Тюнінг правил кореляції та боротьба з втомою від сповіщень**
- 3.9. SIEM і відповідність регуляторним вимогам: звітність compliance**
- 3.10. SIEM у гібридному й мультимарному середовищі**

Середня організація з кількома сотнями робочих станцій генерує десятки тисяч подій безпеки щодня, а деякі, мільйони. Жодна команда аналітиків не здатна прочитати їх усі вручну, і саме тут закінчуються можливості журналювання,

розглянутого в розділі 2, і починається робота SIEM-системи, інструмента, здатного відсіяти шум і виділити ті кілька подій, на які варто звернути увагу людини.

3.1. Визначення SIEM та еволюція технології

SIEM (Security Information and Event Management) поєднує дві історично окремі категорії засобів: SIM (Security Information Management), відповідальний за довгострокове зберігання, звітність і відповідність вимогам, та SEM (Security Event Management), що обробляє події в реальному часі, виконує кореляцію і формує сповіщення. Термін SIEM уперше ввела аналітична компанія Gartner у 2005 році, об'єднавши обидва підходи.

Технологія не стояла на місці, і рисунок 3.1 показує чотири етапи цієї еволюції: від роздільних SIM/SEM у 2000-х через власне SIEM з 2005 року, доповнення SOAR-можливостями автоматизації реагування з 2015 року, і до сучасних хмарних SIEM з вбудованим машинним навчанням та поведінковою аналітикою UEBA у 2020-х.

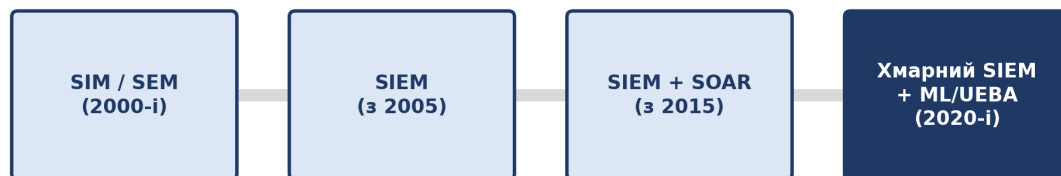


Рис. 3.1. Еволюція технологій моніторингу подій безпеки

Джерело: розроблено автором

Саме тому екзаменаційні питання чинної робочої програми, «Призначення та склад системи моніторингу подій інформаційної безпеки (SIEM)» і «Функції та джерела інформації системи моніторингу подій інформаційної безпеки (SIEM)», заслуговують на окремий розділ, а не на побіжну згадку: без SIEM сучасний SOC-процес просто не існує в тому вигляді, в якому його очікує ринок праці.

3.2. Архітектура SIEM

Попри розмаїття конкретних продуктів (розглянутих далі в підрозділі 3.4 та в розділах 4 і 5), усі SIEM-рішення відтворюють один і той самий узагальнений набір компонентів, показаний на рисунку 3.2.



Рис. 3.2. Узагальнена архітектура SIEM-системи

Джерело: розроблено автором

Рівень збору й нормалізація на рисунку 3.2 повторюють конвеєр із розділу 2: SIEM не замінює журналювання, а надбудовується над ним. Далі подія потрапляє до сховища, а звідти одночасно у три робочі контури: механізм кореляції, що застосовує до потоку подій правила виявлення; керування інцидентами, де пов'язані сповіщення об'єднуються в один робочий елемент для аналітика; і візуалізацію разом із SOAR-плейбуками, які можуть не лише показати дашборд, а й автоматично виконати першу дію реагування.

3.3. Методи кореляції подій

Серце будь-якого SIEM, це механізм кореляції, і саме від обраного методу залежить, що система здатна побачити. Сигнатурна (rule-based) кореляція діє за принципом «якщо сталося N невдалих спроб входу за M хвилин з одного джерела,

сформувані сповіщення»: вона проста, передбачувана і безсила проти нової, ще не описаної атаки. Статистична або поведінкова кореляція шукає відхилення від базової лінії нормальної поведінки. UEBA (User and Entity Behavior Analytics) профілює поведінку користувачів і пристроїв методами машинного навчання, тема, до якої книга повернеться в модулі 3. Зіставлення з Threat Intelligence порівнює індикатори компрометації з потоком подій, забезпечуючи миттєвий контекст: детально цей підхід розглянуто в розділі 8.

Порівняльні сильні й слабкі сторони цих чотирьох методів зведено в таблиці 3.1.

Таблиця 3.1. Порівняння методів кореляції подій у SIEM

Метод кореляції	Принцип роботи	Головна перевага	Головне обмеження
Сигнатурний (rule-based)	Фіксовані правила виду «якщо X, то сповіщення»	Прозорість, передбачуваність, легко пояснити аудиторі	Не бачить атак, для яких немає правила
Статистичний / поведінковий	Порівняння з базовою лінією нормальної поведінки	Виявляє невідомі раніше аномалії	Потребує тривалого «навчання» на чистих даних
UEBA (машинне навчання)	Профілювання поведінки користувачів і сутностей	Виявляє повільні, приховані атаки	Складність пояснення вердикту, ризик упереджень моделі
Зіставлення з Threat Intelligence	Порівняння індикаторів компрометації з подіями	Миттєвий контекст і пріоритизація	Ефективний лише за якісного, свіжого фіда даних

На практиці зрілий SOC не обирає один метод, а комбінує усі чотири: сигнатурні правила ловлять відоме, поведінкова аналітика й UEBA, невідоме, а Threat Intelligence додає контекст до обох.

3.4. Огляд ринку рішень

Ринок SIEM умовно поділяється на три групи. Хмаро-нативні платформи, насамперед Microsoft Sentinel і Google Security Operations, будуються на інфраструктурі хмарного провайдера й не потребують власного апаратного парку. Традиційні платформи з хмарним і локальним розгортанням, зокрема Splunk Enterprise Security, IBM QRadar, Exabeam, LogRhythm, довели ефективність десятиліттями експлуатації у великих організаціях. Рішення на основі відкритого стека, Elastic Security чи орієнтований на менші організації Wazuh, знижують поріг входу за рахунок відкритого коду. В українському контексті організації дедалі частіше обирають саме хмарні SIEM через відсутність потреби підтримувати власну апаратну інфраструктуру та зручну інтеграцію з Microsoft 365 і Entra ID, які й так уже широко використовуються.

Позиції конкретних постачальників у аналітичних звітах на кшталт Gartner Magic Quadrant чи Forrester Wave змінюються щороку, тому ця книга свідомо не наводить рейтингів: перевіряйте їх в актуальних джерелах на момент вивчення теми, а не за друкованим підручником, вік якого може перевищувати вік самого рейтингу.

Окремою дослідницькою темою, суміжною з вибором готового комерційного SIEM, є побудова локальної платформи моніторингу й реагування на базі рішень з відкритим кодом (наприклад, Wazuh) з додатковим шаром П для підтримки SOC-аналітика, як альтернативи комерційним EDR/XDR-рішенням: цей підхід розглянуто в [6] зі списку джерел цього розділу.

Кожна SIEM-платформа має власну мову запитів для написання правил кореляції: KQL у Microsoft Sentinel (розділ 4), SPL у Splunk (розділ 5), власні синтаксиси в Elastic, QRadar та інших рішеннях. Це створює практичну проблему: правило виявлення, написане й перевірене для однієї платформи, доводиться повністю переписувати при переході на іншу, а спільнота дослідників безпеки, що публікує нові правила виявлення атак, змушена або обирати одну платформу, або дублювати роботу кілька разів.

3.5. Критерії вибору SIEM для організації

Обираючи SIEM для конкретної організації, варто зважити принаймні шість критеріїв, зведених у таблиці 3.2.

Таблиця 3.2. Критерії вибору SIEM-рішення для організації

Критерій	На що звернути увагу
Обсяг подій	Очікувані EPS і ГБ/добу, модель ліцензування за обсягом даних, вузлами або користувачами
Розгортання	Хмара, локально чи гібридно; вимоги щодо розташування даних (data residency)
Глибина конекторів	Наявність готових інтеграцій із наявною інфраструктурою організації
Аналітичні правила «з коробки»	Відповідність вбудованих правил матриці MITRE ATT&CK
Сукупна вартість володіння	ТСО на горизонті трьох-п'яти років, а не лише вартість ліцензії за перший рік
Зрілість SOAR	Наявність і якість готових плейбуків автоматизації реагування

3.6. Sigma: універсальна мова опису правил виявлення

Проект **Sigma** вирішує цю проблему як формат опису правил виявлення, незалежний від конкретного постачальника SIEM: правило описується декларативно у форматі YAML, вказуючи умови (яке поле, яке значення, яка логічна умова між ними) без прив'язки до синтаксису конкретної мови запитів. Спеціальний конвертер (sigma-cli) трансліює правило Sigma в цільову мову: KQL для Sentinel, SPL для Splunk, Lucene-запит для Elastic тощо. Це дозволяє організації, по-перше, використовувати відкритий репозиторій готових правил Sigma (SigmaHQ), уже прив'язаних до конкретних технік MITRE ATT&CK, а по-друге, зберегти інвестицію в написані правила навіть у разі майбутньої зміни SIEM-платформи.

3.7. UEBA: аналітика поведінки користувачів і сутностей

Класична кореляція подій за фіксованими правилами (підрозділ 3.3) добре виявляє відомі шаблони атак, але слабо реагує на **інсайдерські загрози та атаки з використанням легітимних, скомпрометованих облікових даних**, коли зловмисник діє під виглядом реального користувача і формально не порушує жодного окремого правила. Для цього класу загроз сучасні SIEM-платформи все частіше включають модуль **UEBA (User and Entity Behavior Analytics)**.

На відміну від правил, що перевіряють окрему подію на відповідність відомому шаблону, UEBA будує індивідуальну поведінкову базову лінію (аналогічно до концепції з підрозділу 1.7, але для облікового запису чи пристрою, а не для системи в цілому) і оцінює сукупність дій користувача за певний період: типовий час входу, типові ресурси, до яких він звертається, типовий обсяг завантажених даних. Аномалія в UEBA - це не окрема підозріла подія, а статистично значуще відхилення поведінки від власної історії користувача або від поведінки його групи однолітків (peer group). Наприклад, головний бухгалтер, що вперше опівночі завантажує весь архів фінансових документів, хоча кожна окрема дія (вхід у систему, читання файлу) сама по собі є цілком легітимною операцією. Детальніше застосування методів машинного навчання до аналізу поведінки розглянуто в розділі 23 модуля 3.

3.8. Тюнінг правил кореляції та боротьба з втомою від сповіщень

Розгортання SIEM «з коробки», з увімкненими вбудованими правилами кореляції з таблиці 3.2, у реальному середовищі організації типово генерує значну кількість хибнопозитивних спрацювань: вбудоване правило написане постачальником для узагальненого середовища й не враховує специфіку конкретної мережі, де, наприклад, певний сканер вразливостей (підрозділ 1.8) чи резервне копіювання регулярно створюють трафік, що формально відповідає шаблону підозрілої активності. Наслідком нетюнінгованого SIEM є та сама проблема **втоми від сповіщень (alert fatigue)**, що вперше згадувалася в підрозділі 1.1 і повторно постає стосовно IDS/IPS у підрозділі 12.5 модуля 2: аналітик, засипаний сотнями малоцінних сповіщень щодня, неминуче починає пропускати серед них і по-справжньому важливі.

Процес тюнінгу SIEM, на відміну від разового налаштування під час впровадження, є безперервним циклом: аналітик регулярно переглядає статистику спрацювань кожного правила, виявляє правила з непропорційно високою часткою підтверджених хибнопозитивних (false positive), додає обґрунтовані винятки (наприклад, виключення відомої IP-адреси власного сканера вразливостей) чи коригує порогові значення, а також, що не менш важливо, аналізує пропущені інциденти (false negative), щоб зрозуміти, якого правила чи джерела даних бракувало. Зрілий SOC веде формальний реєстр таких коригувань правил, аналогічний реєстру змін конфігурації з підрозділу 9.5 модуля 2, щоб зміни в логіці виявлення були простежувані й оборотні.

3.9. SIEM і відповідність регуляторним вимогам: звітність compliance

Підрозділ 3.1 згадав, що історично SIEM виникла зі злиття SIM (орієнтованого на довгострокове зберігання і звітність) та SEM (орієнтованого на кореляцію в реальному часі); функція SIM нікуди не зникла й лишається важливою практичною причиною, з якої організації взагалі впроваджують SIEM, поряд із виявленням атак. Регуляторні стандарти й закони, згадані в підрозділі 1.3 (ISO/IEC 27001, PCI DSS, а для об'єктів критичної інфраструктури, вимоги Держспецзв'язку), як правило, вимагають не просто вести журнали (підрозділ 2.6), а й регулярно документально підтверджувати виконання конкретних контролів безпеки.

Сучасні SIEM-платформи надають для цього готові **дашборди відповідності (compliance dashboards)**: попередньо налаштовані звіти, що зіставляють наявні дані журналювання з переліком вимог конкретного стандарту (наприклад, «чи журналюється кожна зміна прав доступу», «чи зберігаються журнали не менше визначеного терміну») і автоматично формують періодичний звіт для аудитора чи регулятора, замість ручного збору доказів виконання вимог перед кожною перевіркою. Таким чином SIEM виконує подвійну роль в організації: як оперативний інструмент SOC-аналітика (підрозділи 3.2-3.3) і як інструмент документального підтвердження відповідності для служби комплаєнсу, що часто є окремою зацікавленою стороною при виборі платформи (підрозділ 3.5).

3.10. SIEM у гібридному й мультимарному середовищі

Огляд ринку рішень (підрозділ 3.4) розглядав SIEM переважно як єдину платформу, але сучасна організація рідко тримає всю інфраструктуру в одному місці: частина систем лишається в локальному дата-центрі, частина працює в одному чи кількох хмарних провайдерах (розділ 14 модуля 2), і кожне середовище має власний, специфічний формат подій та власний API для їх отримання. Побудова єдиної точки видимості вимагає окремого конектора для кожного джерела (наприклад, локальний syslog-колектор, Azure Activity Log, AWS CloudTrail, Google Cloud Audit Logs), і саме кількість та якість готових конекторів часто стає вирішальним практичним критерієм вибору SIEM-платформи (підрозділ 3.5), більш вагомим за суто технічні характеристики механізму кореляції.

Мультимарна видимість створює додаткову методологічну складність: одна й та сама логічна подія (наприклад, невдала автентифікація) в різних постачальників журналюється в несумісних форматах з різними назвами полів, тому конвеєр нормалізації, вже розглянутий у підрозділі 2.5, для мультимарного середовища

набуває критичного значення, а не лишається технічною деталлю. Без послідовної нормалізації аналітик змушений писати окреме правило кореляції для кожного хмарного провайдера окремо, що суттєво підвищує вартість підтримки платформи порівняно з однорідним локальним середовищем.

Практикум 3. Порівняльний аналіз SIEM-рішень

Мета: навчитися формувати обґрунтований вибір SIEM для умовної організації, спираючись на критерії з таблиці 3.2.

Хід роботи:

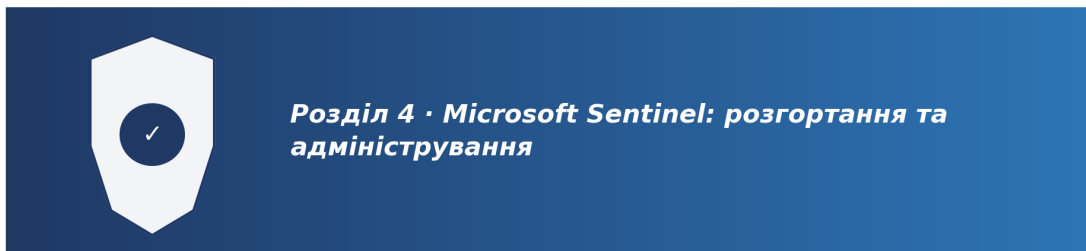
1. отримати варіант умовної організації (розмір, галузь, наявна інфраструктура: локальна, хмарна чи гібридна);
2. побудувати порівняльну таблицю трьох SIEM-рішень (мінімум одне хмарне, одне традиційне, одне з відкритим кодом) за критеріями таблиці 3.2;
3. розрахувати орієнтовний обсяг подій (EPS) організації на основі кількості вузлів і типових джерел;
4. письмово обґрунтувати вибір (одна-дві сторінки), явно пояснивши, яким критеріям надано пріоритет і чому.

Контрольні питання до розділу 3

1. Що означає аббревіатура SIEM і з яких двох категорій засобів вона історично утворилася?
2. Опишіть чотири етапи еволюції технології, зображені на рисунку 3.1.
3. Перелічіть основні компоненти архітектури SIEM із рисунка 3.2 та поясніть роль кожного.
4. Порівняйте чотири методи кореляції подій із таблиці 3.1: який з них найкраще виявляє невідому раніше атаку і чому?
5. За якими критеріями обирають SIEM-рішення для організації?
6. Чому підручник свідомо не наводить рейтингу постачальників SIEM?
7. Яка різниця між SIM і SEM як історичними попередниками SIEM?
8. Перелічіть основні функціональні рівні архітектури SIEM.
9. Що таке правило кореляції (correlation rule) і наведіть приклад простого правила.
10. Чим кореляція на основі машинного навчання відрізняється від кореляції на основі фіксованих правил?
11. Що таке SOAR і як він пов'язаний із SIEM?

12. Які фактори варто врахувати при виборі SIEM-рішення для організації, крім вартості ліцензії?
13. Чим хмарний SIEM відрізняється від локально розгорнутого (on-premise)?
14. Що таке «час утримання» (retention) подій у SIEM і як він впливає на вартість рішення?
15. Наведіть приклад ситуації, коли надто «чутливе» правило кореляції шкодить роботі SOC.
16. Наведіть приклад, коли кореляція подій із кількох різних джерел виявляє атаку, непомітну за жодним джерелом окремо.

РОЗДІЛ 4. MICROSOFT SENTINEL: РОЗГОРТАННЯ ТА АДМІНІСТРУВАННЯ



Зміст розділу

- 4.1. Місце Sentinel в екосистемі безпеки Microsoft***
- 4.2. Основні компоненти Sentinel***
- 4.3. Основи мови запитів KQL (Kusto Query Language)***
- 4.4. Підключення джерел даних***
- 4.5. Аналітичні правила та автоматизація реагування***
- 4.6. Керування вартістю: рівні збереження даних***
- 4.7. UEBA та інтеграція Threat Intelligence в Sentinel***
- 4.8. Workbooks: візуалізація та інтерактивна звітність***
- 4.9. Керування доступом і багатоклієнтська архітектура Sentinel***
- 4.10. Content Hub і готові рішення для типових сценаріїв виявлення***

Розділ 3 описав SIEM як клас систем в загальному вигляді. Тепер час перейти від теорії до конкретної платформи, з якою випускник з високою ймовірністю зіткнеться на першому робочому місці: більшість українських організацій, що вже

перейшли на Microsoft 365 і Entra ID, обирають як SIEM саме Microsoft Sentinel, оскільки інтеграція з наявною інфраструктурою виявляється природною, а не додатковим проєктом.

4.1. Місце Sentinel в екосистемі безпеки Microsoft

Microsoft Sentinel — хмаро-нативна платформа класу SIEM+SOAR, що працює на базі Azure Log Analytics Workspace. На відміну від традиційних SIEM, Sentinel не потребує розгортання власної апаратної інфраструктури для зберігання й обробки подій: масштабування, зберігання даних та обчислювальні ресурси надаються хмарною платформою Azure. Sentinel інтегрується з продуктами лінійки Microsoft Defender (Defender for Endpoint, Defender for Cloud, Defender for Identity, Defender for Office 365), утворюючи разом з ними XDR-екосистему (Extended Detection and Response): якщо SIEM збирає й корелює події з усієї організації, XDR фокусується на глибокій, попередньо оснащений кореляції всередині родини продуктів одного виробника.

4.2. Основні компоненти Sentinel

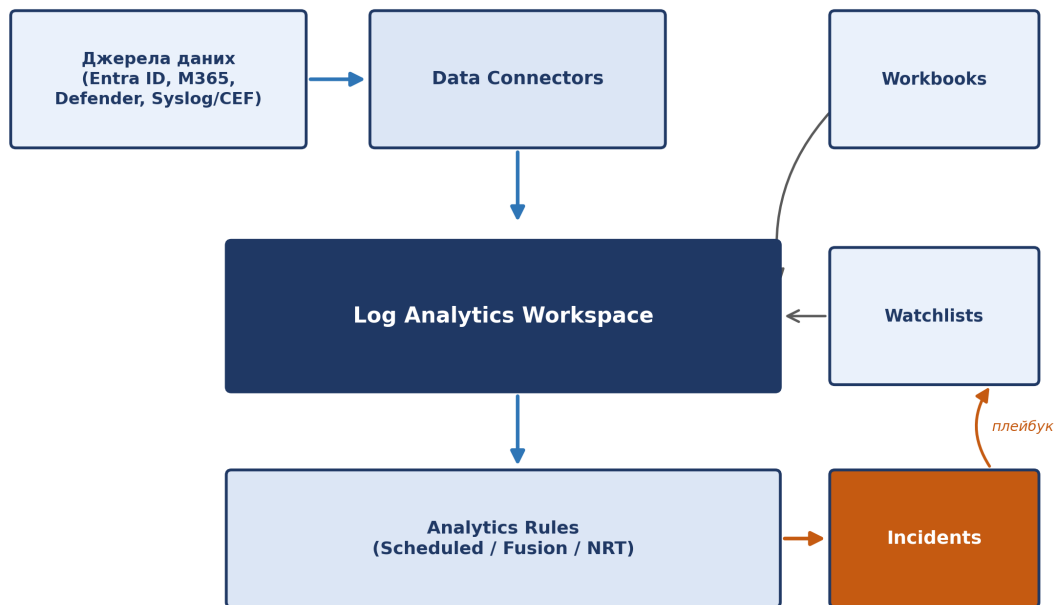
Компоненти платформи природно групуються за роллю в конвеєрі «дані → виявлення → реагування», що узагальнено в таблиці 4.1.

Таблиця 4.1. Основні компоненти Microsoft Sentinel

Компонент	Призначення
Log Analytics Workspace	сховище даних, на якому працює Sentinel; кожна таблиця відповідає типу подій (SigninLogs, SecurityEvent, CommonSecurityLog)
Data Connectors	вбудовані інтеграції з джерелами: Entra ID, Microsoft 365, продукти Defender, Syslog/CEF, конектори до AWS/GCP
Analytics Rules	логіка виявлення: за розкладом (scheduled), на основі ML-кореляції (Fusion), майже в реальному часі (NRT)
Incidents	об'єднання пов'язаних сповіщень в один робочий елемент для аналітика
Workbooks	інтерактивні дашборди
Playbooks	сценарії автоматичного реагування на базі Azure Logic Apps (SOAR-функціональність)

Watchlists	списки для збагачення подій контекстом (наприклад, перелік привілейованих облікових записів)
Threat Intelligence	розділ для імпорту й перегляду індикаторів компрометації (докладніше — розділ 8.6)

Рисунок 4.1 показує, як ці компоненти складаються в єдиний потік: дані з джерел через конектори потрапляють до робочої області, звідти аналітичні правила породжують інциденти, а плейбуки автоматично реагують на них.



*Рис. 4.1. Архітектура Microsoft Sentinel
Джерело: розроблено автором*

4.3. Основи мови запитів KQL (Kusto Query Language)

KQL — декларативна мова запитів до табличних даних, спільна для Log Analytics, Sentinel, Azure Monitor та Microsoft Defender XDR (Advanced Hunting). Синтаксично запит KQL — це конвеєр послідовних операторів, з'єднаних символом `|`, і в цьому він концептуально близький до SPL, з якою студент ознайомиться в розділі 5.

Приклад 1. Пошук облікових записів з великою кількістю невдалих спроб входу за останню добу:

```
SigninLogs
|           where           TimeGenerated           >           ago(1d)
|           where           ResultType           !=           "0"
| summarize FailedAttempts = count() by UserPrincipalName, IPAddress
|           where           FailedAttempts           >           5
| sort by FailedAttempts desc
```

Приклад 2. Виявлення входу з незвично великої кількості країн для одного користувача (спрощена логіка):

```
SigninLogs
|           where           TimeGenerated           >           ago(30d)
| summarize Countries = make_set(Location) by UserPrincipalName
| where array_length(Countries) > 3
```

Ключові оператори KQL, які має опанувати студент: where (фільтрація), summarize (агрегація), project/extend (вибір і обчислення полів), join (об'єднання таблиць), make_set/make_list (згортання значень), bin (групування за інтервалами часу).

4.4. Підключення джерел даних

Загальний порядок підключення джерела: обрати конектор у Content Hub → надати необхідні дозволи (найчастіше через реєстрацію застосунку в Microsoft Entra ID або системну керовану ідентичність) → активувати збір потрібних типів подій → перевірити надходження даних у відповідну таблицю Log Analytics Workspace тестовим KQL-запитом за зразком розділу 4.3.

4.5. Аналітичні правила та автоматизація реагування

Scheduled query rules виконують KQL-запит за розкладом (наприклад, кожні 5–15 хвилин) і створюють сповіщення при виконанні умови. **Fusion**, вбудований механізм кореляції на основі машинного навчання, поєднує слабкі сигнали з різних джерел у сценарії атак із високою достовірністю: наприклад, «незвичний вхід» разом із «підозрілою активністю PowerShell» на тому самому обліковому записі протягом короткого проміжку часу створюють значно переконливіший сигнал, ніж кожна подія окремо.

Playbook (на основі Azure Logic Apps) автоматизує типові дії реагування: збагачення інциденту даними Threat Intelligence, сповіщення відповідального в

месенджері, тимчасове блокування облікового запису, створення заявки в системі service desk. Саме зв'язка «Analytics Rule формує інцидент → Playbook автоматично реагує» перетворює Sentinel із системи звітності на інструмент активного захисту, зображений на рисунку 4.1.

4.6. Керування вартістю: рівні збереження даних

Оскільки Sentinel тарифікується за обсяг даних, що надходять до Log Analytics Workspace (підрозділ 2.8 розглядає цю проблему загалом), Microsoft пропонує кілька рівнів збереження, що дозволяють гнучко балансувати вартість і доступність даних для пошуку. **Analytics Logs** - основний, найдорожчий рівень із повною індексацією й миттєвим доступом для KQL-запитів та аналітичних правил. **Basic Logs** - дешевший рівень для типів подій з високим обсягом, але низькою цінністю для активної кореляції (наприклад, детальні мережеві журнали), з обмеженою функціональністю пошуку. **Archive** дозволяє зберігати дані значно довше (для виконання вимог зберігання) за мінімальну вартість, без можливості прямого запиту, лише з попереднім відновленням (rehydration) потрібного інтервалу перед аналізом. Правильний розподіл типів подій між цими рівнями, а не бездумне направлення всього потоку в Analytics Logs, є основним важелем контролю вартості володіння Sentinel (TCO, таблиця 3.2).

4.7. UEBA та інтеграція Threat Intelligence в Sentinel

Концепція аналітики поведінки користувачів, введена в підрозділі 3.7, реалізована в Sentinel окремим модулем **Entity Behavior**, який будує профілі активності для користувачів і хостів на основі підключених журналів (SigninLogs, SecurityEvent) і виявляє відхилення від них, не потребуючи ручного написання окремого KQL-правила для кожного можливого сценарію. Профіль сутності також агрегує всі інциденти, пов'язані з конкретним користувачем чи хостом, у єдину часову шкалу, що суттєво прискорює розслідування.

Окремим блоком є конектори **Threat Intelligence**, які дозволяють імпортувати індикатори компрометації (розділ 8) із зовнішніх платформ (TIP) або безпосередньо за стандартом STIX/TAXII, і зіставляти їх із наявними подіями через вбудовані аналітичні правила «TI map». Такий збіг, наприклад IP-адреса з журналу мережевого трафіку, що збігається з відомою шкідливою адресою з фіда Threat Intelligence, автоматично формує інцидент високої достовірності, оскільки спирається не на евристику, а на підтверджений зовнішній індикатор.

4.8. Workbooks: візуалізація та інтерактивна звітність

Таблиця 4.1 назвала Workbook як інтерактивний дашборд; розглянемо цей компонент детальніше, оскільки саме він перетворює сирі результати KQL-запитів (підрозділ 4.3) на форму, придатну для щоденної роботи аналітика й для звітності перед керівництвом. **Workbook** поєднує кілька візуальних елементів (графіки, таблиці, карти, текстові блоки з поясненнями) в одну інтерактивну сторінку, де кожен елемент живиться результатом окремого KQL-запиту й може оновлюватися в реальному часі. На відміну від статичного звіту, Workbook підтримує параметри, керовані самим користувачем (наприклад, випадний список для вибору часового діапазону чи конкретного джерела даних), що дозволяє одному й тому самому Workbook слугувати і оперативним дашбордом чергового аналітика, і інструментом для щомісячної презентації метрик служби безпеки, згаданих у підрозділі 1.4 (MTTD, MTTR, обсяг оброблених інцидентів).

Sentinel постачається з галереєю готових шаблонів Workbook під конкретні джерела даних (наприклад, окремий Workbook для аналізу подій Entra ID чи для огляду стану всіх підключених конекторів), які можна використовувати як є або взяти за основу для власного, специфічного під задачі конкретної організації дашборда.

4.9. Керування доступом і багатоклієнтська архітектура Sentinel

Принцип найменших привілеїв, введений у підрозділі 9.4 модуля 2, у Sentinel реалізується через вбудовані ролі Azure RBAC, специфічні для платформи: **Microsoft Sentinel Reader** (лише перегляд даних та інцидентів, без права змінювати конфігурацію), **Microsoft Sentinel Responder** (додатково дозволяє працювати з інцидентами: призначати, змінювати статус, коментувати), **Microsoft Sentinel Contributor** (повний доступ до створення й редагування аналітичних правил, Workbook, конекторів). Такий розподіл дозволяє, наприклад, надати аналітику першої лінії (L1) лише роль Responder, залишивши створення й тюнінг самих правил кореляції (підрозділ 3.8) за досвідченішим інженером з роллю Contributor.

Для організацій, що керують безпекою кількох окремих клієнтів одночасно, найчастіше для постачальників послуг MSSP, згаданих у підрозділі 1.10, Sentinel підтримує **багатоклієнтську архітектуру (multi-tenancy)** через **Azure Lighthouse**: аналітик постачальника отримує делеговані права на робочі області Sentinel кожного клієнта, керуючи ними з єдиної консолі, без необхідності мати окремий обліковий запис у кожному орендарі (tenant) Azure клієнта. Це технічно реалізує саме той

гібридний сценарій розподілу L1/L2/L3 між власною командою й зовнішнім постачальником, що обговорювався в підрозділі 1.10.

4.10. Content Hub і готові рішення для типових сценаріїв виявлення

Написання аналітичного правила кореляції з нуля (підрозділ 4.5) для кожного нового джерела даних чи типового сценарію атаки є трудомістким і не завжди виправданим, коли йдеться про вже добре вивчену загрозу. **Content Hub**, каталог готових рішень (Solutions) у складі Sentinel, надає для багатьох поширених продуктів (наприклад, конкретного класу міжмережевого екрана чи хмарного сервісу) готовий пакет: набір аналітичних правил, Workbook для візуалізації (підрозділ 4.8), парсери нормалізації подій і, за потреби, playbook автоматизованого реагування, що встановлюється практично одним кліком замість тижнів самостійної розробки.

Практична цінність такого підходу для аналітика полягає не лише в економії часу, а й у якості: готове рішення, як правило, розроблене й перевірене постачальником продукту чи спільнотою на реальних інцидентах, тоді як самостійно написане з нуля правило кореляції на ранніх етапах експлуатації частіше страждає від хибнопозитивних спрацювань (підрозділ 3.8), поки не буде відповідно доналаштоване під конкретне середовище організації. Водночас готове рішення не звільняє від обов'язкового тюнінгу під власну інфраструктуру: воно є відправною точкою, а не остаточним налаштуванням.

Практикум 4. Розгортання робочої області Sentinel і базового правила кореляції

Мета: отримати навички розгортання Sentinel, підключення джерела даних і написання аналітичного правила.

Хід роботи:

1. розгорнути Log Analytics Workspace і активувати на ньому Microsoft Sentinel (у навчальній хмарній підписці/пісочниці);
2. підключити щонайменше одне джерело даних (наприклад, Azure Activity або тестовий Syslog-конектор);
3. написати й перевірити KQL-запит за прикладом розділу 4.3;
4. на основі запиту створити scheduled analytics rule з визначенням порогу і рівня критичності;
5. згенерувати тестову подію, що відповідає умові правила, переконатися у створенні інциденту;

6. (опційно) підключити найпростіший Playbook сповіщення.

Примітка для викладача. Конкретні назви розділів інтерфейсу Azure можуть змінюватися між релізами: перед заняттям слід звірити актуальний вигляд порталу з документацією Microsoft Learn.

Контрольні питання до розділу 4

1. На якому сховищі даних побудовано Microsoft Sentinel?
2. Перелічіть основні компоненти Sentinel із таблиці 4.1 та поясніть роль кожного.
3. Чим Fusion відрізняється від звичайного scheduled query rule?
4. Яку роль виконують Playbooks і на якій технології вони базуються?
5. Напишіть KQL-запит, що виводить кількість подій входу за останню годину, згруповану за користувачем.
6. Що таке Log Analytics Workspace і яку роль він відіграє в архітектурі Sentinel (рисунок 4.1)?
7. Чим Data Connector відрізняється від Analytics Rule?
8. Опишіть призначення Watchlists у Sentinel.
9. Що таке Fusion-правило і на чому базується його логіка виявлення?
10. Наведіть базову структуру KQL-запиту (from-where-project/summarize).
11. Чим NRT (Near Real-Time) правило відрізняється від звичайного scheduled query rule за частотою виконання?
12. Що таке Workbook у Sentinel і для чого він використовується?
13. Назвіть щонайменше три типи джерел даних, які можна підключити до Sentinel.
14. Чим ліцензування Sentinel (оплата за обсяг даних) відрізняється від традиційного ліцензування SIEM за кількістю вузлів?
15. Опишіть, як інцидент, створений Analytics Rule, потрапляє до черги реагування аналітика.
16. Яку перевагу дає інтеграція Sentinel з рештою продуктів екосистеми Microsoft (розділ 4.1)?

РОЗДІЛ 5. SPLUNK: АРХІТЕКТУРА, ПОШУК ПОДІЙ, ДАШБОРДИ



**Розділ 5 · Splunk: архітектура, пошук
подій, дашборди**

Зміст розділу

- 5.1. Архітектура Splunk**
- 5.2. Дані у Splunk: індекси, sourcetype, поля**
- 5.3. Мова пошуку SPL (Search Processing Language)**
- 5.4. Common Information Model і Splunk Enterprise Security**
- 5.5. Дашборди та візуалізація**
- 5.6. Ліцензування та управління обсягом даних**
- 5.7. Splunk SOAR: автоматизація реагування**
- 5.8. Кластеризація та відмовостійкість Splunk**
- 5.9. Керування доступом і ролями у Splunk**
- 5.10. Machine Learning Toolkit: вбудовані засоби аналітики Splunk**

Якщо Sentinel з розділу 4 природно обирають організації, що вже живуть в екосистемі Microsoft, то Splunk лишається орієнтиром для великих, часто гетерогенних інфраструктур, де SIEM експлуатують десятиліттями і де важлива свобода розгортання: на власних серверах, у хмарі або в гібридній конфігурації. Логіка платформи інша, ніж у Sentinel, але концептуально паралельна, і порівняння цих двох підходів само по собі є цінною навчальною вправою.

5.1. Архітектура Splunk

Архітектура Splunk будується навколо трьох ролей, показаних на рисунку 5.1, до яких за потреби додаються два допоміжні компоненти адміністрування.

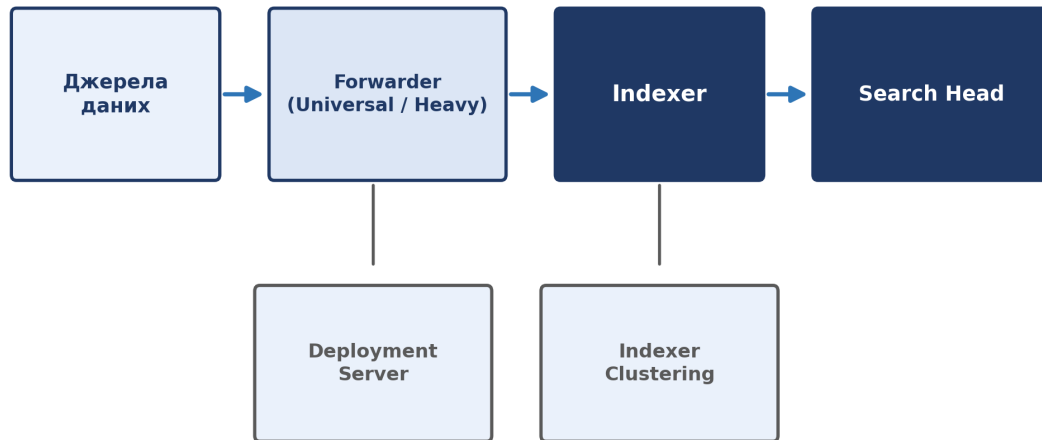


Рис. 5.1. Архітектура Splunk
Джерело: розроблено автором

Forwarder встановлюється на джерелі даних і передає події до індексатора: Universal Forwarder виконує лише пересилання, тоді як Heavy Forwarder додатково здійснює попередню обробку і фільтрацію ще до передавання. **Indexer** приймає, парсить й індексує дані, зберігаючи їх у вигляді індексів. **Search Head** надає інтерфейс пошуку й аналітики, розподіляючи запити між індексаторами; search head clustering застосовується на великих інсталяціях для відмовостійкості й балансування навантаження. Два допоміжні компоненти на рисунку 5.1, **Deployment Server** (централізоване керування конфігурацією форвардерів) і **Indexer Clustering** (реплікація даних між індексаторами), відповідають, відповідно, за керованість і відмовостійкість інсталяції в міру її зростання.

5.2. Дані у Splunk: індекси, sourcetype, поля

Подія (event) — одиничний запис у Splunk. Індекс (index) — іменований контейнер, у якому фізично зберігаються події (наприклад, окремі індекси для мережесих подій, подій автентифікації, подій застосунків). Sourcetype визначає формат і спосіб парсингу подій конкретного джерела. Розрізняють **index time** (обробка при записі: парсинг, визначення часової мітки) і **search time** (обчислення додаткових полів під час виконання пошукового запиту): розуміння цієї відмінності

важливе для продуктивності великих інсталяцій, оскільки перенесення обробки на час пошуку економить час запису, але сповільнює кожен запит, що звертається до цих полів.

5.3. Мова пошуку SPL (Search Processing Language)

Подібно до KQL з розділу 4, SPL побудована як конвеєр команд, з'єднаних символом `|`.

Приклад 1. Топ-10 джерел за кількістю заблокованих міжмережовим екраном з'єднань за добу:

```
index=firewall          action=blocked          earliest=-24h
| stats count by src_ip
| sort -count
| head 10
```

Приклад 2. Побудова часового ряду невдалих автентифікацій:

```
index=auth              action=failure
| timechart span=1h count by user
```

Основні команди SPL, необхідні студенту, зведено в таблиці 5.1.

Таблиця 5.1. Основні команди мови SPL

Команда	Призначення
search	базовий пошук подій за умовою
stats	агрегація (кількість, сума, середнє тощо) за групувальними полями
table	вибір конкретних полів для виведення
eval	обчислювані поля на основі виразу
timechart	побудова часового ряду з групуванням
transaction	групування пов'язаних подій в одну сесію
rex	вилучення полів регулярними виразами

5.4. Common Information Model і Splunk Enterprise Security

CIM (Common Information Model), функціональний аналог ASIM у Sentinel з розділу 4, це набір стандартизованих схем полів (наприклад, моделі Authentication, Network Traffic, Malware), що дає змогу писати правила кореляції незалежно від конкретного sourcetype джерела. **Splunk Enterprise Security (ES)**, надбудова класу

SIEM над платформою Splunk, додає notable events (аналог інцидентів), кореляційні пошуки (correlation searches) за розкладом і ризик-орієнтоване сповіщення (**Risk-Based Alerting, RBA**): накопичення «балів ризику» для об'єкта (користувача чи хоста) із кількох джерел замість генерації окремого сповіщення на кожну подію, що суттєво знижує «втому від сповіщень» (alert fatigue), про яку вже йшлося в розділі 1.

5.5. Дашборди та візуалізація

Splunk Dashboard Studio дає змогу будувати інтерактивні панелі на основі збережених SPL-пошуків: часові ряди, теплові карти, географічні карти за IP-геолокацією, таблиці топ-N. Дашборди SOC зазвичай включають динаміку кількості інцидентів за критичністю, топ джерел атак і статус покриття джерел подій, тобто health check конвеєра журналювання з розділу 2.

5.6. Ліцензування та управління обсягом даних

Традиційна модель ліцензування Splunk історично базувалася на обсязі даних, що індексуються за добу (ГБ/добу), хоча останніми роками постачальник дедалі активніше пропонує альтернативні моделі (за кількістю віртуальних процесорів, за робочим навантаженням у хмарній версії Splunk Cloud). Незалежно від конкретної моделі ліцензування, керування обсягом даних лишається практичною задачею адміністратора: кожен індекс у Splunk фізично складається з **buckets** (кошиків), що проходять життєвий цикл станів **hot** (щойно записані, активно поповнюються) → **warm** (закриті для запису, але доступні для пошуку) → **cold** (переміщені на дешевше, повільніше сховище) → **frozen** (архівовані або видалені після завершення терміну зберігання, визначеного політикою підрозділу 2.6). Правильне налаштування розміру й терміну життя кожного стану кошика під конкретні вимоги організації (швидкість пошуку проти вартості зберігання) є одним із головних інструментів адміністрування великої інсталяції Splunk, концептуально паралельним до рівнів збереження Sentinel з підрозділу 4.6.

5.7. Splunk SOAR: автоматизація реагування

Аналогічно до Playbooks у Sentinel (підрозділ 4.5), Splunk пропонує окремий продукт для автоматизації реагування, **Splunk SOAR** (раніше відомий як Phantom). SOAR-платформа дозволяє описати послідовність дій реагування у вигляді візуального або програмованого **playbook**: збагачення інциденту даними з зовнішніх джерел (репутація IP-адреси, дані Threat Intelligence з розділу 8), автоматичну

ізоляцію скомпрометованого хоста через інтеграцію з засобами кінцевого захисту (EDR), створення заявки в системі service desk, сповіщення відповідальної команди. Ключова відмінність SOAR від простого сповіщення полягає в **оркестрації**: платформа взаємодіє з десятками зовнішніх систем через готові інтеграції (apps), виконуючи послідовність дій, які вручну зайняли б в аналітика значно більше часу, ніж триває сам playbook.

5.8. Кластеризація та відмовостійкість Splunk

Підрозділ 5.1 згадав indexer clustering і search head clustering як допоміжні компоненти; розглянемо, яку задачу вони насправді вирішують у міру зростання інсталяції. **Indexer clustering** реплікує кожен вхідний блок даних (bucket, підрозділ 5.6) на кілька індексаторів одночасно, за налаштованим **коефіцієнтом реплікації (replication factor)**: якщо один індексатор виходить з ладу, дані лишаються доступними на інших копіях, а сам кластер автоматично перепризначає роль головного вузла для непошкоджених копій. Окремо від реплікації даних задається **коефіцієнт пошуку (search factor)**, кількість копій кожного bucket, які підтримуються в одразу готовому до пошуку стані, тоді як решта копій зберігаються в необробленому вигляді й вимагають додаткового часу на відновлення пошукової придатності в разі відмови.

Search head clustering, натомість, розподіляє навантаження й забезпечує відмовостійкість не для зберігання даних, а для самого шару пошуку та інтерфейсу аналітика: кілька search head об'єднуються в кластер із спільною конфігурацією (збережені пошуки, дашборди, права доступу), і користувач може підключитися до будь-якого вузла кластера, отримуючи однаковий результат, а відмова одного вузла не позбавляє аналітиків можливості працювати. Разом ці два механізми реалізують для Splunk ту саму мету відмовостійкості й безперервності служби, яку RPO/RTO (підрозділ 11.5 модуля 2) задають для базових служб організації в цілому.

5.9. Керування доступом і ролями у Splunk

Розмежування прав у Splunk будується на системі **ролей (roles)** і **можливостей (capabilities)**: роль є іменованим набором можливостей (наприклад, право виконувати пошук, право редагувати збережені пошуки, право керувати користувачами), а кожному користувачу призначається одна або кілька ролей, аналогічно до вбудованих ролей Sentinel з підрозділу 4.9. Вбудовані ролі Splunk (admin, power, user) охоплюють типові сценарії, але зрілі інсталяції зазвичай

визначають власні, вужче окреслені ролі відповідно до принципу найменших привілеїв (підрозділ 9.4 модуля 2).

Специфічною для Splunk можливістю є обмеження доступу не лише за функціональністю (що дозволено робити), а й за даними (що дозволено бачити): роль можна прив'язати до конкретного переліку індексів (підрозділ 5.2), тому, наприклад, аналітик підрозділу фінансової безпеки може отримати доступ до індексу подій платіжної системи, лишаючись без доступу до індексу подій кадрової системи, навіть маючи однакові функціональні права пошуку в обох випадках. Такий контроль на рівні даних, а не лише на рівні функцій інтерфейсу, є важливою відмінністю адміністрування великої SIEM-платформи від адміністрування типового бізнес-застосунку.

5.10. Machine Learning Toolkit: вбудовані засоби аналітики Splunk

Поряд із запитамі SPL для явно сформульованих умов (підрозділ 5.3), Splunk надає окремий додаток, **Machine Learning Toolkit (MLTK)**, що дозволяє застосовувати алгоритми машинного навчання (кластеризація, виявлення аномалій, прогнозування) безпосередньо до даних, уже проіндексованих платформою, без потреби вивантажувати їх у зовнішнє середовище аналізу. Типовий сценарій використання: побудова моделі очікуваного обсягу мережевого трафіку певного сегмента за історичними даними, з подальшим автоматичним сповіщенням, коли фактичний обсяг суттєво відхиляється від прогнозованого діапазону, що технічно є прикладною реалізацією підходів, розглянутих поглиблено в розділі 23 модуля 3.

MLTK орієнтований передусім на аналітика, обізнаного з основами статистики, а не на професійного інженера машинного навчання: інтерфейс надає готові майстри для типових задач (виявлення викидів, прогнозування часових рядів) без необхідності писати власний код на Python, хоча можливість розширення власними алгоритмами так само є. Практична цінність такого вбудованого інструментарію для SOC полягає в скороченні шляху від виявленої в даних закономірності до працюючого сповіщення: аналітик може перевірити гіпотезу й розгорнути модель у межах однієї платформи, не чекаючи окремого циклу розробки зовнішньою командою дата-сайнс.

Практикум 5. Побудова дашборду моніторингу автентифікацій у Splunk

Мета: отримати навички побудови SPL-запитів і візуалізації результатів.

Хід роботи:

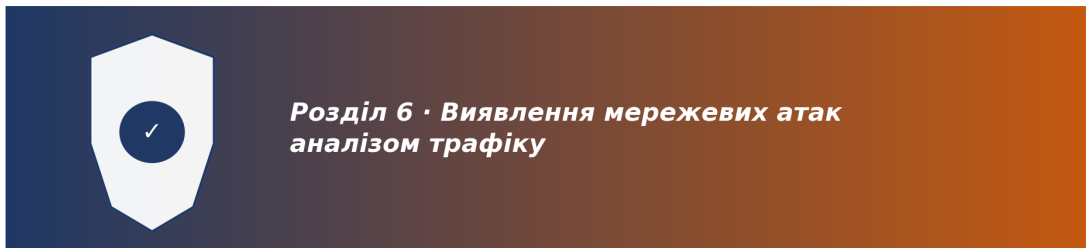
1. завантажити в тестовий індекс Splunk вибірку подій автентифікації (навчальний набір даних);
2. побудувати SPL-запит підрахунку невдалих спроб входу за користувачем і джерельною IP-адресою;
3. виявити облікові записи з аномально великою кількістю невдалих спроб (потенційний brute-force);
4. оформити результати у вигляді дашборду з часовим рядом і таблицею топ-10 підозрілих джерел;
5. створити на основі запиту alert (сповіщення), що спрацьовує при перевищенні порогового значення.

Контрольні питання до розділу 5

1. Опишіть призначення трьох основних компонентів архітектури Splunk (Forwarder, Indexer, Search Head).
2. Чим index time відрізняється від search time?
3. Напишіть SPL-запит, що виводить кількість подій за годину для індексу vpn.
4. Що таке Risk-Based Alerting і яку проблему SOC він вирішує?
5. Наведіть приклад CIM-моделі та поясніть, навіщо вона потрібна.
6. Яку роль виконує Forwarder у архітектурі Splunk (рисунок 5.1) і чим Universal Forwarder відрізняється від Heavy Forwarder?
7. Що таке sourcetype і навіщо він потрібен під час індексування даних?
8. Опишіть базову структуру SPL-запиту з командами search, stats і table.
9. Чим індексований (indexed) час відрізняється від часу події (event time)?
10. Що таке Splunk Enterprise Security і яку функцію він додає до базового Splunk?
11. Навіщо потрібна Common Information Model (CIM) при побудові дашбордів і кореляційних пошуків?
12. Що таке Deployment Server і яку задачу він вирішує в розподіленій архітектурі Splunk?
13. Опишіть призначення Indexer Clustering і яку проблему воно вирішує.

14. Чим дашборд (dashboard) відрізняється від збереженого пошуку (saved search) у Splunk?
15. Наведіть приклад використання Risk-Based Alerting для зниження кількості хибнопозитивних інцидентів.
16. Порівняйте архітектурні підходи Splunk і Microsoft Sentinel: у чому головна відмінність моделі розгортання?

РОЗДІЛ 6. ВИЯВЛЕННЯ МЕРЕЖЕВИХ АТАК ШЛЯХОМ АНАЛІЗУ ТРАФІКУ



Зміст розділу

- 6.1. Методи аналізу мережевого трафіку**
- 6.2. Системи виявлення вторгнень мережевого рівня**
- 6.3. Типові мережеві атаки та їхні ознаки в трафіку**
- 6.4. Моніторинг за допомогою NetFlow/IPFIX**
- 6.5. Аналіз зашифрованого трафіку без розшифрування**
- 6.6. Виявлення й пом'якшення DDoS-атак**
- 6.7. Honeyrot та обманні технології (deception technology)**
- 6.8. Виявлення латерального руху за допомогою аналізу потоків**
- 6.9. DNS-моніторинг як індикатор компрометації**

SIEM, розглянутий у розділах 3-5, працює з подіями, які вже хтось згенерував: журнал застосунку, спрацювання антивірусу, запис Active Directory. Але частина атак ніколи не залишає такого зручного сліду: вони видні тільки в самому мережевому трафіку, до того як досягнуть кінцевого вузла. Цей розділ повертається на рівень нижче, до пакетів і потоків, звідки й починається шлях більшості вторгнень.

6.1. Методи аналізу мережевого трафіку

Рисунок 6.1 систематизує чотири методи аналізу трафіку за принципом виявлення разом з альтернативним розрізненням за обсягом даних, що аналізується.

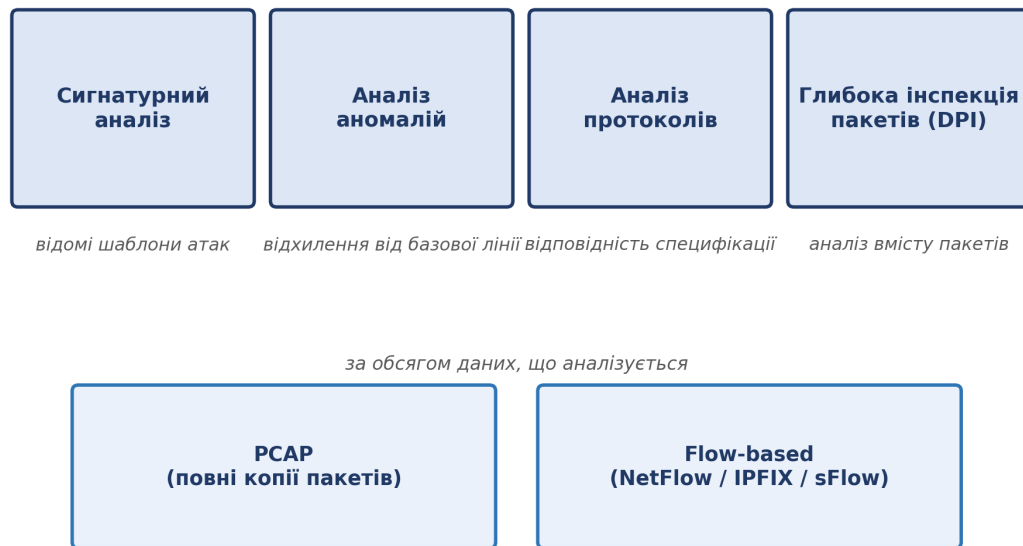


Рис. 6.1. Методи аналізу мережевого трафіку
Джерело: розроблено автором

Сигнатурний аналіз (signature-based) зіставляє пакети чи потоки з базою відомих шаблонів атак: точність на відомих загрозах висока, але метод безсилий проти нових, ще не описаних (zero-day) атак. **Аналіз аномалій (anomaly-based)** виявляє відхилення від базової лінії нормальної поведінки мережі, тобто обсягу трафіку, кількості з'єднань, розподілу протоколів: здатний виявляти невідомі атаки ціною більшої кількості хибнопозитивних спрацювань. **Аналіз протоколів (protocol analysis, stateful inspection)** перевіряє відповідність трафіку специфікації протоколу, виявляючи аномальні або сфальсифіковані пакети. **Глибока інспекція пакетів (Deep Packet Inspection, DPI)** аналізує не лише заголовки, а й вміст пакетів.

Незалежно від обраного методу, дані для аналізу можна збирати у двох принципово різних форматах, показаних у нижній частині рисунка 6.1: **аналіз повних копій пакетів (PCAP)**, максимально детальний, але вимогливий до обчислювальних ресурсів і місця зберігання, та **потіковий аналіз (flow-based, NetFlow/IPFIX/sFlow)**, агреговані метадані про з'єднання без вмісту пакетів, значно

економніші й придатні для моніторингу магістральних каналів. До другого підходу книга повернеться в підрозділі 6.4.

6.2. Системи виявлення вторгнень мережевого рівня

- **Snort**, класична сигнатурна IDS/IPS з відкритим кодом, правила якої описують шаблони підозрілого трафіку;
- **Suricata**, багатопотокова IDS/IPS/NSM з підтримкою протокового аналізу на прикладному рівні та сумісністю з форматом правил Snort;
- **Zeek** (колишній Bro), система мережевого моніторингу безпеки (Network Security Monitor), що не тільки генерує сповіщення за сигнатурами, скільки формує детальні структуровані журнали про кожну мережеву транзакцію: з'єднання, DNS-запити, HTTP-сесії, TLS-сертифікати. Ці журнали, по суті, є ще одним джерелом подій для конвеєра з розділу 2 і зручним матеріалом для подальшої SIEM-кореляції з розділу 3.

6.3. Типові мережеві атаки та їхні ознаки в трафіку

Уміння розпізнати атаку за характерним «почерком» у трафіку, це навичка, яку неможливо здобути лише з визначень: таблиця 6.1 зводить п'ять поширених атак разом з ознаками, за якими їх варто шукати під час практикуму.

Таблиця 6.1. Типові мережеві атаки та їхні ознаки в трафіку

Атака	Типова ознака в трафіку
Сканування портів	велика кількість коротких з'єднань до послідовних портів одного вузла за короткий час
DDoS	різке аномальне зростання обсягу трафіку/кількості з'єднань з багатьох джерел до одного вузла
ARP-spoofing	дублювання MAC-адрес для однієї IP-адреси, аномальна кількість ARP-відповідей
DNS-tunneling	аномально довгі або часті DNS-запити, нетипові типи записів, висока ентропія імен піддоменів

Комунікація з C2-сервером (Command and Control)	періодичні («маячкові», beaconing) з'єднання до одного зовнішнього вузла з регулярним інтервалом
---	--

6.4. Моніторинг за допомогою NetFlow/IPFIX

NetFlow (розроблений Cisco) та відкритий стандарт IPFIX дають змогу моніторити обсяги й характер трафіку на магістральних каналах без повного захоплення пакетів: збираються метадані кожного потоку (5-tuple: IP джерела й приймача, порти, протокол, а також обсяг байтів/пакетів і тривалість). Це основне джерело для виявлення DDoS-атак і аномалій обсягу трафіку у великих мережах, де повний PCAP-аналіз, згаданий у підрозділі 6.1, технічно неможливий через обсяг даних.

6.5. Аналіз зашифрованого трафіку без розшифрування

Дедалі більша частка мережевого трафіку, включно зі шкідливим, шифрується протоколом TLS (детальніше розділ 13 модуля 2), що унеможливорює традиційну глибоку інспекцію вмісту пакетів (DPI, підрозділ 6.1) без повного розшифрування трафіку, яке саме по собі створює додаткові ризики приватності й продуктивності. Тому дослідники безпеки розробили методи аналізу метаданих TLS-сесії, що не потребують розшифрування корисного навантаження.

Найпоширенішим є **TLS-фінгерпринтинг (JA3/JA3S)**: під час встановлення TLS-з'єднання клієнт і сервер обмінюються повідомленнями ClientHello і ServerHello, параметри яких (підтримувані версії протоколу, набори шифрів, розширення, у певному порядку) утворюють характерний «відбиток», що обчислюється як хеш цих параметрів. Оскільки конкретні бібліотеки й версії шкідливого програмного забезпечення часто використовують нестандартні або застарілі TLS-бібліотеки, їхній JA3-відбиток клієнта відрізняється від відбитка легітимного браузера, навіть якщо решта з'єднання повністю зашифрована. Аналітик SOC може зіставляти спостережувані JA3-відбитки з базами відомих шкідливих відбитків, аналогічно до сигнатурного аналізу з підрозділу 6.1, не порушуючи конфіденційність самого зашифрованого змісту з'єднання. Додатковими ознаками, доступними без розшифрування, є розмір і час надсилання пакетів (packet size/timing analysis) та довжина й формат сертифіката сервера.

6.6. Виявлення й пом'якшення DDoS-атак

Атака на відмову в обслуговуванні через розподілену мережу вузлів (DDoS, Distributed Denial of Service), коротко згадана в таблиці 6.1, заслуговує на окремий розгляд через масштаб можливих наслідків: на відміну від більшості атак цього розділу, метою DDoS є не крадіжка даних, а унеможливлення доступу легітимних користувачів до сервісу. Розрізняють DDoS-атаки на **мережевому й транспортному рівнях** (об'ємні атаки, що вичерпують пропускну здатність каналу чи ресурси мережевого обладнання, наприклад SYN flood, UDP flood, атаки посилення через DNS/NTP) і на **прикладному рівні** (менші за обсягом, але цілеспрямовані атаки на конкретний ресурсоємний застосунок, наприклад HTTP flood на сторінку пошуку бази даних).

Виявлення DDoS спирається передусім на аналіз аномалій (підрозділ 6.1) відносно базової лінії нормального трафіку (підрозділ 1.7): різке зростання обсягу з'єднань, кількості унікальних джерел або частки незавершених TCP-рукоштовань (SYN без ACK) є характерними ранніми ознаками. Пом'якшення (mitigation) типово поєднує кілька рубежів: фільтрацію на рівні провайдера чи спеціалізованого хмарного сервісу захисту (scrubbing center) для об'ємних атак, що перевищують пропускну здатність власного каналу організації; обмеження швидкості запитів (rate limiting) і виклик-відповідь (challenge-response, наприклад CAPTCHA) на рівні застосунку; заздалегідь підготовлений план реагування (аналогічний Incident Response Plan з розділу 10 модуля 2), оскільки під час самої атаки час на прийняття рішень украй обмежений.

6.7. Honeypot та обманні технології (deception technology)

Усі методи, розглянуті в попередніх підрозділах, аналізують трафік, що прямує до реальних, продуктивних ресурсів організації, і за визначенням реагують уже на активну спробу атаки. **Honeypot (пастка)** пропонує протилежний за логікою підхід: свідомо розгорнутий, привабливий на вигляд, але фальшивий ресурс (сервер, обліковий запис, файл), що не бере участі в жодному легітимному бізнес-процесі організації. Оскільки жоден легітимний користувач чи процес не має причини звертатися до honeypot, будь-яка взаємодія з ним, за визначенням, є підозрілою і майже гарантовано вказує на розвідувальну активність зловмисника, що вже проник у мережу, задовго до того, як він дістанеться реального цільового ресурсу.

Розрізняють **honeypot низької взаємодії** (лише емулює окремі мережеві служби на рівні, достатньому для реєстрації спроби з'єднання, простий у розгортанні

й супроводі, але легше розпізнається досвідченим зловмисником) і **honeypot високої взаємодії** (повноцінна, хоч і ізольована система, що дозволяє спостерігати за реальними діями зловмисника всередині неї, цінніша для розуміння його тактик, але вимагає значно ретельнішої ізоляції, щоб компрометація самого honeypot не стала плацдармом для подальшої атаки на реальну інфраструктуру). Мережа з кількох honeypot, поєднана для збору узагальненої статистики про тактики атак, називається **honeynet**. Технологія **deception** розширює цю ідею за межі окремих пасток до розсіяних по всій мережі фальшивих облікових даних, з'єднань і записів у службі каталогів (підрозділ 11.1 модуля 2), що суттєво підвищує ймовірність виявити зловмисника на етапі розвідки мережі зсередини (lateral movement, підрозділ 6.8), ще до того, як він дістанеться справжньої цілі.

6.8. Виявлення латерального руху за допомогою аналізу потоків

Підрозділ 6.4 представив NetFlow/IPFIX як засіб моніторингу обсягу трафіку на магістральних каналах; розглянемо окремий, специфічний сценарій його застосування - виявлення **латерального (бічного) руху**, тобто переміщення зловмисника від початково скомпрометованого вузла до інших вузлів усередині мережі в пошуках цінніших цілей. На відміну від DDoS чи сканування портів (таблиця 6.1), латеральний рух зазвичай не створює аномально великого обсягу трафіку, а тому погано виявляється методами, орієнтованими на об'єм; натомість він залишає характерний **структурний** слід у графі з'єднань.

Аналіз NetFlow дає змогу побудувати граф взаємодії вузлів мережі за певний період і виявити відхилення від очікуваної топології спілкування: наприклад, робоча станція бухгалтера, що в нормі звертається лише до поштового сервера й кількох вебресурсів, а тепер послідовно встановлює з'єднання з портом 445 (SMB) чи 3389 (RDP) до кількох інших робочих станцій підряд, демонструє шаблон, типовий саме для розвідки мережі зсередини (техніка Lateral Movement матриці MITRE ATT&CK, підрозділ 8.4), а не для звичайної офісної роботи. Такий структурний аналіз потоків доповнює мікросегментацію мережі (підрозділ 13.6 модуля 2) як другий, виявляючий (detective), а не лише запобіжний (preventive) рубіж проти поширення атаки всередині вже частково скомпрометованої мережі.

6.9. DNS-моніторинг як індикатор компрометації

Окремим, часто недооціненим джерелом даних для виявлення атак є журнал запитів служби доменних імен (DNS), розглянутої детальніше з погляду

адміністрування в підрозділі 11.2 модуля 2. Практично будь-яка сучасна шкідлива програма, від найпростішого завантажувача до складного інструменту командування й контролю (C2), потребує розв'язання доменного імені перед встановленням з'єднання із зовнішнім сервером, тому потік DNS-запитів організації є одним із найбагатших джерел непрямих індикаторів компрометації, порівнюваним за цінністю з аналізом мережевого трафіку (підрозділ 6.1).

Типові ознаки, на які орієнтується DNS-моніторинг: запити до домену, зареєстрованого лише кілька годин чи днів тому (частий індикатор інфраструктури зловмисника, оскільки легітимні домени, як правило, існують значно довше); аномально високий обсяг унікальних піддоменів одного домену за короткий час, що є ознакою тунелювання даних через DNS (DNS tunneling), коли зловмисник кодує викрадені дані в самих доменних іменах запитів, обходячи традиційний контроль каналів передавання, вже розглянутий у розділі 7; і запити до доменів, що входять до відомих списків індикаторів компрометації (підрозділ 8.3). На відміну від аналізу повного мережевого трафіку, DNS-журнали компактні за обсягом і рідко шифруються (за винятком DNS over HTTPS/TLS), що робить їх практичним, недорогим першим рубежем виявлення навіть для організацій з обмеженими ресурсами моніторингу.

Практикум 6. Аналіз мережевого трафіку для виявлення атак

Мета: набути навичок виявлення ознак атаки за аналізом захопленого трафіку.

Хід роботи:

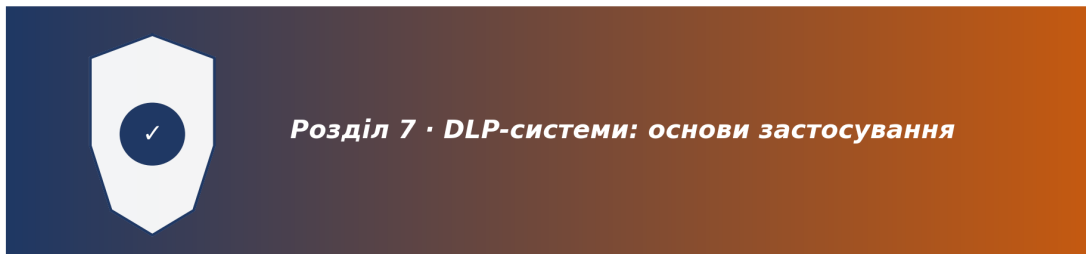
1. отримати навчальний PCAP-файл, що містить суміш легітимного трафіку та трафіку сканування портів;
2. проаналізувати файл інструментом аналізу трафіку (Wireshark для ручного аналізу окремих пакетів, Zeek — для отримання структурованих журналів з'єднань);
3. виявити вузол-джерело сканування за ознаками з таблиці 6.1;
4. написати правило Suricata/Snort, що виявляє аналогічну активність у майбутньому;
5. оформити висновок: які поля трафіку виявилися вирішальними для детекції.

Контрольні питання до розділу 6

1. Чим сигнатурний аналіз трафіку відрізняється від аналізу аномалій?
2. Чим PCAP-аналіз відрізняється від потокового (flow-based) аналізу і коли застосовується кожен з них?
3. Опишіть відмінність підходу Zeek від класичних сигнатурних IDS (Snort/Suricata).
4. Які ознаки в трафіку свідчать про DNS-tunneling (таблиця 6.1)?
5. Що таке «beaconing» і про що він може свідчити?
6. Порівняйте PCAP-аналіз і flow-based аналіз за обсягом даних, що зберігається, і глибиною деталізації (рисунок 6.1).
7. Що таке глибока інспекція пакетів (DPI) і чим вона відрізняється від аналізу заголовків пакета?
8. Наведіть приклад ознаки сканування портів (port scan) у мережевому трафіку.
9. Чим атака типу «людина посередині» (MITM) проявляється в аналізі трафіку?
10. Що таке NetFlow і які поля типово містить запис потоку (flow record)?
11. Чим IPFIX відрізняється від класичного NetFlow v5/v9?
12. Наведіть приклад того, як аномально велика кількість DNS-запитів до одного домену може свідчити про DNS-tunneling.

13. Чим Zeek (Bro) відрізняється концептуально від сигнатурних IDS на кшталт Snort?
14. Яку перевагу дає поєднання flow-based моніторингу з повними копіями пакетів (PCAP) для окремих сесій?
15. Опишіть практичний сценарій виявлення атаки перебору пароля (brute-force) за аналізом мережевого трафіку.
16. Чим система виявлення вторгнень мережевого рівня (розділ 6) відрізняється від повноцінного IDS/IPS-рішення, детально розглянутого в розділі 12 модуля 2?

РОЗДІЛ 7. DLP-СИСТЕМИ: ОСНОВИ ЗАСТОСУВАННЯ



Зміст розділу

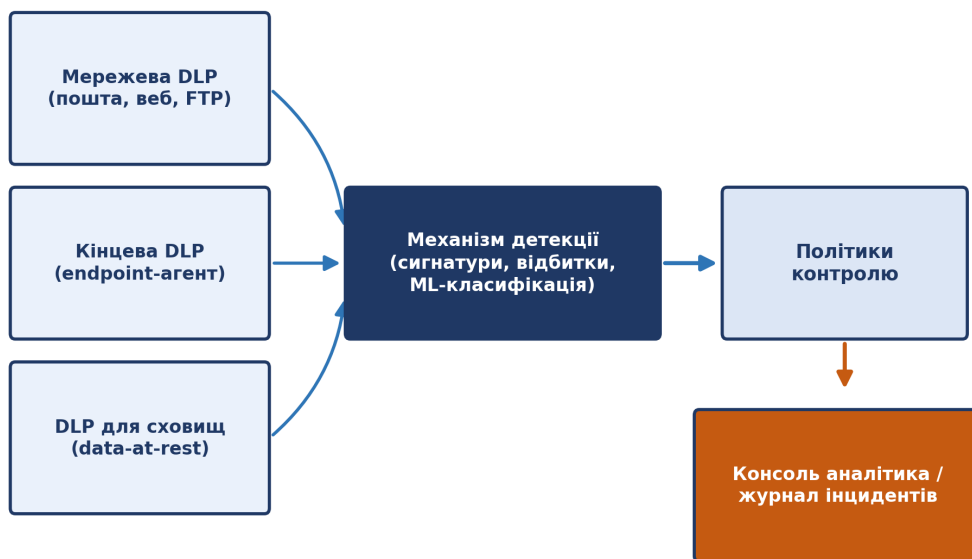
- 7.1. Призначення та архітектура DLP-систем**
- 7.2. Методи детекції контенту**
- 7.3. Канали контролю**
- 7.4. Огляд SearchInform DLP**
- 7.5. Розробка політик DLP і юридичні аспекти моніторингу співробітників**
- 7.6. DLP у хмарному середовищі та для віддалених працівників**
- 7.7. Метрики ефективності програми DLP**
- 7.8. Класифікація даних як передумова ефективної DLP**
- 7.9. Виявлення інсайдерських загроз: поєднання DLP і поведінкової аналітики**
- 7.10. Контроль хмарних застосунків: CASB як доповнення до традиційної DLP**

Усі інструменти, розглянуті дотепер, від SNMP-моніторингу до SIEM-кореляції, відповідають на запитання «хто, коли і звідки діяв у системі». DLP-системи ставлять інше запитання: «що саме перетинає межу організації і чи має воно на це право». Це єдиний клас засобів у цьому модулі, що аналізує не поведінку, а

зміст інформації, і саме тому вимагає окремого, обережнішого розгляду, включно з юридичним виміром.

7.1. Призначення та архітектура DLP-систем

DLP (Data Loss Prevention/Data Leak Prevention) — клас засобів, призначених для запобігання несанкціонованому передаванню чутливої інформації за межі контрольованого периметра організації. На відміну від засобів периметрового захисту (міжмережевих екранів), DLP аналізує не адресу з'єднання, а **зміст** інформації, що передається. Архітектурно розрізняють мережеву DLP (network DLP), контроль трафіку на периметрі: пошта, веб, FTP; кінцеву DLP (endpoint DLP), агент на робочій станції, що контролює локальні канали: знімні носії, друк, буфер обміну, локальні застосунки; і DLP для сховищ даних (discovery/data-at-rest DLP), сканування файлових сховищ, СУБД і хмарних сховищ на предмет наявності й неналежного розташування чутливих даних. Рисунок 7.1 показує, як ці три джерела сходяться до єдиного механізму детекції, а звідти, через застосування політик, до консолі аналітика.



*Рис. 7.1. Архітектура DLP-системи
Джерело: розроблено автором*

7.2. Методи детекції контенту

Ефективність механізму детекції на рисунку 7.1 визначається тим, які методи в ньому реалізовано; таблиця 7.1 порівнює чотири основні підходи.

Таблиця 7.1. Методи детекції контенту в DLP-системах

Метод	Принцип	Типове застосування
Сигнатурний пошук	ключові слова, регулярні вирази	номери карток, ІПН, паспортні дані
«Цифрові відбитки» (fingerprinting)	точне порівняння з еталонним документом або фрагментом	захист конкретних відомих документів
Контекстний і статистичний аналіз	тип документа, гриф, відправник/отримувач	політики за категорією документообігу
Класифікація на основі машинного навчання	визначення категорії за змістом без точного збігу з еталоном	виявлення нових, раніше не бачених чутливих документів

7.3. Канали контролю

Типова DLP-система контролює корпоративну й вебпошту, вебтрафік (завантаження на файлообмінники, соціальні мережі), знімні носії (USB), друк документів, буфер обміну, месенджери, а в розширеній конфігурації, також аудіо- та відеопотоки робочого середовища.

7.4. Огляд SearchInform DLP

SearchInform — програмний комплекс класу DLP, прямо згаданий у чинній робочій програмі дисципліни (практичне заняття «Налаштування програмного комплексу SearchInform для контролю мережі» і лабораторна робота «Встановлення та налагодження програмного комплексу SearchInform»). Загальна архітектура рішень цього класу типово включає агенти на кінцевих станціях, що перехоплюють дії користувача за контрольованими каналами; сервер(и) обробки та індексації перехопленого контенту; консоль адміністратора чи аналітика, у якій налаштовуються політики контролю та переглядаються інциденти; і модуль звітності та розслідування, що дає змогу відтворити хронологію дій користувача, аналогічно до того, як SIEM з розділу 3 відтворює хронологію дій зловмисника.

Примітка. Конкретні кроки встановлення, назви модулів і вигляд консолі можуть відрізнятися між версіями продукту: при виконанні практикуму слід користуватися чинною документацією виробника, отриманою разом з ліцензією чи навчальним доступом.

7.5. Розробка політик DLP і юридичні аспекти моніторингу співробітників

Впровадження DLP пов'язане не лише з технічними, а й з юридичними ризиками: контроль кореспонденції та дій працівника зачіпає його права. В організації, що впроваджує DLP, необхідно закріпити право на моніторинг у внутрішніх політиках (положення про інформаційну безпеку, політика прийнятного використання, Acceptable Use Policy), з якими працівник ознайомлюється під підпис; дотримуватися вимог законодавства про захист персональних даних щодо цілей і пропорційності обробки даних; розмежовувати робочі та приватні канали зв'язку наскільки це технічно можливо; і визначити коло осіб, які мають доступ до результатів перехоплення, забезпечивши журналювання їхніх власних дій, тобто контроль контролерів, той самий принцип розділення обов'язків, що вже згадувався в підрозділі 2.6.

Технічна ефективність DLP значною мірою залежить від якості політик: надто широкі правила спричиняють шквал хибнопозитивних спрацювань і «втому» аналітика (та сама проблема alert fatigue, що й у SIEM), надто вузькі, пропускають реальні витоки.

7.6. DLP у хмарному середовищі та для віддалених працівників

Класична модель DLP, описана в підрозділах 7.1-7.3, спроектована для трафіку, що проходить через контрольований периметр організації: корпоративний поштовий шлюз, проксі-сервер вихідного вебтрафіку. Масове поширення хмарних сервісів (SaaS-застосунків для документообігу й комунікації, розглянутих у контексті журналювання в підрозділі 2.7) і віддаленої роботи руйнує це припущення: значна частина трафіку співробітника взагалі не проходить через периметр організації, а йде напяму з домашньої мережі до хмарного сервісу.

Відповіддю на цей виклик є клас рішень **CASB (Cloud Access Security Broker)**, що діє як проміжний контрольний пункт між користувачами та хмарними застосунками незалежно від мережі, з якої йде з'єднання: реалізується або в режимі проксі (forward/reverse проху, трафік користувача проходить через CASB), або через прямі API-інтеграції з хмарним провайдером (наприклад Microsoft 365, Google Workspace), що дає змогу сканувати вже збережений у хмарі контент без перехоплення трафіку взагалі. Для кінцевих станцій поза периметром роль мережевої DLP частково бере на себе посилена endpoint DLP, здатна застосовувати політику незалежно від того, до якої мережі підключений ноутбук.

7.7. Метрики ефективності програми DLP

Впровадження DLP не закінчується розгортанням, описаним у практикумі й лабораторній роботі нижче: як і будь-яка функція SOC, програма DLP потребує регулярної оцінки за вимірюваними показниками, інакше ризик «встановили й забули» (аналогічний ризик уже згадувався щодо тюнінгу правил кореляції в підрозділі 3.4). Типові метрики включають частку хибнопозитивних спрацювань відносно загальної кількості інцидентів (чим вона вища, тим швидше настає «втома» аналітика, підрозділ 7.5); середній час обробки одного інциденту аналітиком, від моменту спрацювання політики до ухвалення рішення; частку каналів передавання даних, охоплених хоча б однією політикою контролю (coverage), оскільки неконтрольований канал зводить нанівець захист решти; і, за наявності ретроспективних даних, кількість підтверджених витоків, запобігти яким вдалося саме завдяки DLP, що є найпереконливішим аргументом при обґрунтуванні бюджету програми перед керівництвом.

7.8. Класифікація даних як передумова ефективної DLP

Таблиця 7.1 назвала «гриф» документа одним із критеріїв контекстного аналізу, але сама наявність грифа не з'являється автоматично: її повинна забезпечити окрема, здебільшого організаційна, а не суто технічна практика **класифікації даних**. Без класифікації DLP-система змушена або застосовувати одну й ту саму політику до всієї інформації, що передається (неминуче або надто жорстко, або надто м'яко для більшості документів), або покладатися виключно на розпізнавання за форматом даних (номер картки, ППН), пропускаючи чутливі документи, що не містять подібних формально розпізнаваних шаблонів, наприклад стратегічний план розвитку компанії чи текст ще не опублікованого патенту.

Класифікацію проводять за схемою рівнів чутливості (наприклад: публічна, внутрішня, конфіденційна, таємна), визначеною політикою інформаційної безпеки організації (підрозділ 9.2 модуля 2), і технічно реалізують через **мітки чутливості (sensitivity labels)**: метадані, вбудовані в сам документ, що подорожують разом із ним незалежно від того, де він зберігається чи копіюється. Мітки може призначати автор документа вручну під час створення, або система може призначати їх автоматично на основі виявленого вмісту (тими самими методами детекції з підрозділу 7.2). Такі мітки суттєво спрощують і саму роботу DLP: політика може посилатися просто на «документи з міткою Конфіденційно», а не намагатися заново розпізнавати конфіденційність вмісту щоразу під час передавання.

7.9. Виявлення інсайдерських загроз: поєднання DLP і поведінкової аналітики

Класична DLP-політика (підрозділи 7.2-7.3) реагує на окрему подію передавання даних, що порушує правило, незалежно від того, хто її вчинив і за яких обставин. Але найнебезпечніші випадки витоку даних часто передують не одна яскрава подія, а поступова зміна поведінки конкретного співробітника: працівник, що подав заяву про звільнення, раптово починає масово завантажувати файли з корпоративного сховища; обліковий запис, що зазвичай працює лише в робочі години, починає активність вночі. Жодна з цих дій окремо не обов'язково порушує жорстку політику DLP, але їхня сукупність утворює характерний профіль ризику.

Клас рішень **Insider Threat Management (ITM)** поєднує події DLP із поведінковою аналітикою UEBA, розглянутою в підрозділі 3.7, для оцінки сукупного ризику конкретного співробітника, а не лише окремої події передавання даних: до факторів ризику додають кадрові події (наприклад, підтверджена дата звільнення, отримана з системи кадрового обліку), відхилення від типового робочого графіка, доступ до ресурсів поза звичайною сферою обов'язків. Практичне впровадження такого поєднання вимагає особливо ретельного дотримання юридичних меж, викладених у підрозділі 7.5, оскільки побудова поведінкового профілю конкретної людини є суттєво чутливішим втручанням у приватність, ніж контроль окремого каналу передавання даних.

7.10. Контроль хмарних застосунків: CASB як доповнення до традиційної DLP

Підрозділ 7.6 розглянув DLP у хмарному середовищі на рівні загальних принципів; окремим спеціалізованим класом рішень саме для цього завдання є **CASB (Cloud Access Security Broker)**, проміжний шар контролю, що розташовується між користувачами організації та хмарними SaaS-застосунками (наприклад, корпоративною поштою, файловим сховищем, CRM-системою). CASB працює у двох принципово різних режимах: **прямий проксі (forward proxy)**, що перехоплює трафік до хмарного сервісу в реальному часі й може заблокувати заборонену дію ще до її виконання, і **режим API-інтеграції (API-based)**, що ретроспективно сканує вже наявний вміст хмарного сховища через офіційний API постачальника, виявляючи порушення постфактум, без затримки трафіку користувача.

Практична цінність CASB для організації, що активно використовує SaaS-застосунки, полягає в закритті прогалини видимості, яку традиційна мережева DLP (підрозділ 7.1), орієнтована на контроль каналів усередині корпоративної мережі, не

покриває: співробітник, що завантажує конфіденційний документ безпосередньо з домашнього комп'ютера у хмарне сховище через вебінтерфейс постачальника, обходить корпоративний периметр повністю, а CASB, працюючи на рівні самого хмарного застосунку, а не мережевого каналу, залишається здатним виявити таку дію. Багато сучасних CASB-рішень також виконують функцію виявлення «тіньового ІТ» (Shadow IT): несанкціонованого використання співробітниками хмарних сервісів, не затверджених службою безпеки організації.

Практикум 7. Налаштування політики DLP

Мета: отримати навички формування та тестування політики контролю каналу передавання даних.

Хід роботи:

1. визначити категорію чутливої інформації для контролю (наприклад, документи з грифом «Комерційна таємниця» або персональні дані клієнтів);
2. сформулювати правило детекції (за ключовими словами/регулярним виразом або цифровим відбитком еталонного документа, таблиця 7.1);
3. застосувати правило до каналу «знімні носії» і каналу «вебпошта»;
4. провести контрольне тестування: спробувати передати тестовий документ, що підпадає під правило, і документ, що не підпадає;
5. проаналізувати журнал інцидентів, оцінити частку хибнопозитивних спрацювань і за потреби скоригувати правило.

Лабораторна робота. Встановлення та налагодження програмного комплексу SearchInform

Відповідно до навчально-методичної картки дисципліни, лабораторна робота передбачає розгортання серверної частини та агента SearchInform у тестовому середовищі, підключення агента до сервера, перевірку надходження перехопленого трафіку в консоль та первинне налаштування щонайменше двох каналів контролю. Детальний покроковий регламент лабораторної роботи оформлюється викладачем окремими методичними вказівками з огляду на версію продукту, доступну кафедрі на момент проведення заняття.

Контрольні питання до розділу 7

1. Чим мережева DLP відрізняється від кінцевої (endpoint) DLP?
2. Які методи детекції контенту застосовуються в сучасних DLP-системах (таблиця 7.1)?
3. Перелічіть основні компоненти архітектури SearchInform.
4. Які юридичні вимоги слід врахувати при впровадженні DLP-контролю дій співробітників?
5. Чому надто широкі політики DLP шкідливі для ефективності SOC?

6. Назвіть три типи DLP-систем за точкою розгортання (мережева, кінцева, для сховищ) і коротко опишіть кожен.
7. Що таке «цифровий відбиток» документа (digital fingerprint) і як він використовується для детекції?
8. Чим детекція за ключовими словами відрізняється від детекції на основі машинного навчання?
9. Перелічіть щонайменше чотири канали, які повинна контролювати повноцінна політика DLP.
10. Чим режим лише журналювання (monitor-only) відрізняється від режиму активного блокування в політиці DLP?
11. Опишіть архітектуру «агент-сервер», яку використовує SearchInform.
12. Чому надто широка політика DLP шкодить не лише зручності користувачів, а й ефективності роботи SOC?
13. Яким чином DLP-система може виявити спробу передавання лише фрагмента, а не всього конфіденційного документа?
14. Порівняйте задачі адміністрування DLP-системи і адміністрування IDS/IPS: що в них спільного щодо тюнінгу хибних спрацювань?
15. Наведіть приклад того, як DLP-контроль повинен бути узгоджений із внутрішньою політикою інформаційної безпеки організації.
16. Чому канал знімних носіїв (USB) залишається одним із найважче контрольованих для DLP-системи?

РОЗДІЛ 8. THREAT INTELLIGENCE: ЗБІР, АНАЛІЗ, ІНТЕГРАЦІЯ З SIEM



Зміст розділу

8.1. Поняття та рівні Threat Intelligence

8.2. Життєвий цикл кіберрозвідки (Intelligence Cycle)

8.3. Джерела Threat Intelligence

8.4. Стандарти обміну та класифікації

8.5. Платформи Threat Intelligence

8.6. Інтеграція Threat Intelligence з SIEM

8.7. Оцінювання якості та достовірності Threat Intelligence

8.8. Threat Hunting: проактивне використання Threat Intelligence

8.9. Атрибуція атак: угруповання загроз та їхні профілі

8.10. Обмін розвідданими на національному й галузевому рівні в Україні

8.11. MITRE ATT&CK як спільна мова Threat Intelligence і SOC

Кожен розділ цього модуля, від SNMP-моніторингу до DLP, працював з даними, породженими всередині власної організації. Але зловмисник, що атакує сьогодні, майже напевно вже атакував когось іншого вчора, залишивши сліди, якими той інший може поділитися. Threat Intelligence, це останній, завершальний шар модуля: перетворення чужого досвіду атак на дані, що заздалегідь підвищують пильність власної системи моніторингу.

8.1. Поняття та рівні Threat Intelligence

Threat Intelligence (кіберрозвідка загроз) — оброблена й проаналізована інформація про наявні або потенційні загрози, що дає змогу організації приймати обґрунтовані рішення щодо захисту. Прийнято виокремлювати чотири рівні: **стратегічний**, тенденції загроз для керівництва, наприклад зростання активності певного типу шахрайства в галузі; **оперативний**, інформація про конкретні кампанії та наміри зловмисників, корисна для середньої ланки управління безпекою; **тактичний**, тактики, техніки й процедури (TTP) зловмисників, корисні для побудови правил виявлення в SIEM (розділ 3); і **технічний**, конкретні індикатори компрометації (IOC): IP-адреси, домени, хеші файлів, той рівень, з яким найчастіше працює SOC-аналітик безпосередньо.

8.2. Життєвий цикл кіберрозвідки (Intelligence Cycle)

Кіберрозвідка, як і будь-яка розвідувальна діяльність, це не одноразова дія, а замкнений цикл, показаний на рисунку 8.1.

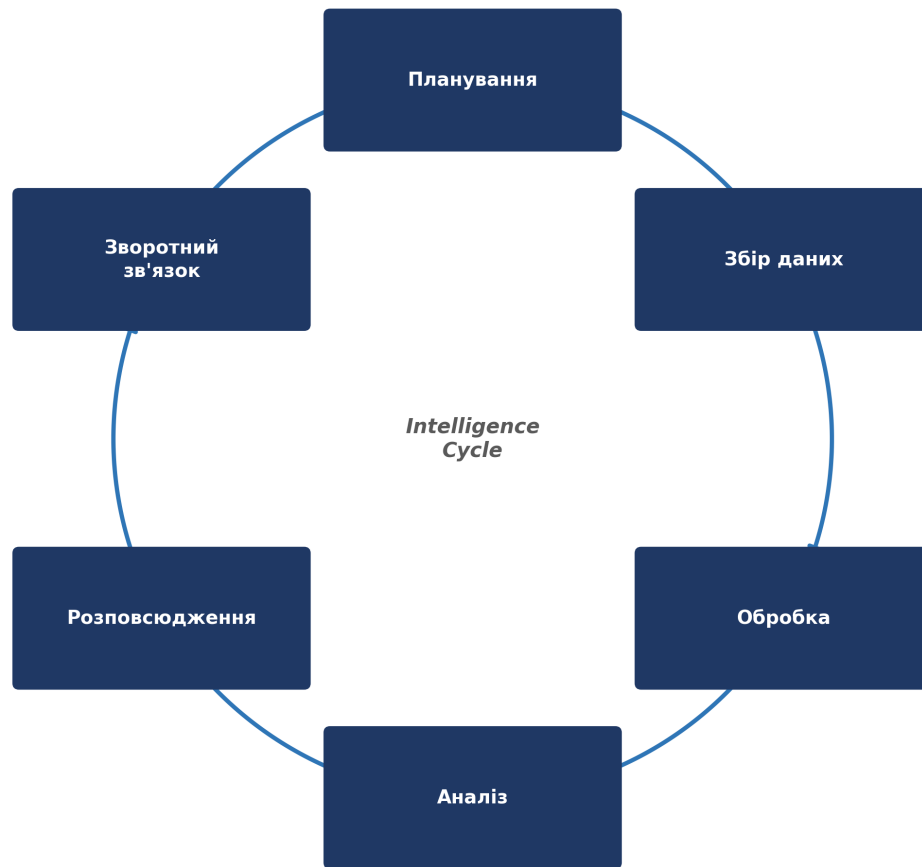


Рис. 8.1. Життєвий цикл кіберрозвідки (Intelligence Cycle)

Джерело: розроблено автором

Планування та постановка завдання визначають, яка саме інформація потрібна і кому. Збір даних наповнює цикл сирим матеріалом з джерел, розглянутих у підрозділі 8.3. Обробка приводить зібране до єдиного формату: нормалізацію й дедублікацію, у чомусь аналогічну нормалізації подій із розділу 2. Аналіз встановлює зв'язки між фрагментами інформації та оцінює їхню достовірність. Розповсюдження доводить готовий результат до зацікавлених підрозділів, насамперед SOC. Зворотний зв'язок оцінює, наскільки корисним виявився результат, і уточнює завдання наступного циклу, замикаючи коло на етапі планування.

8.3. Джерела Threat Intelligence

- **OSINT (Open Source Intelligence):** публічні джерела, блоги дослідників безпеки, звіти вендорів, публікації в соціальних мережах;
- **CERT-UA та галузеві ISAC/ISAO:** національні та галузеві центри обміну інформацією про загрози;
- **комерційні фіди:** платні потоки індикаторів від спеціалізованих постачальників;
- **спільноти обміну:** платформи на кшталт MISP (підрозділ 8.5), де організації діляться індикаторами на добровільній основі.

8.4. Стандарти обміну та класифікації

STIX (Structured Threat Information Expression), поточна версія 2.1, це стандартизована мова опису об'єктів кіберрозвідки (індикатор, зловмисне ПЗ, кампанія, загрозливий суб'єкт) та зв'язків між ними. **TAXII (Trusted Automated Exchange of Intelligence Information)** — протокол автоматизованого обміну даними у форматі STIX між платформами: якщо STIX описує, «що» передається, то TAXII визначає, «як» це передається. **MITRE ATT&CK**, вже згадана в розділі 3 матриця тактик і технік зловмисників (Reconnaissance, Initial Access, Execution, Persistence, Privilege Escalation, Lateral Movement, Exfiltration тощо), є спільною мовою опису поведінки атаки для аналітика SOC, розробника правил SIEM і команди Threat Intelligence одночасно. Окремо варто розрізняти **ІОС проти ІОА**: Indicator of Compromise, технічний слід атаки, що вже відбулася (хеш, IP, домен), на відміну від Indicator of Attack, поведінкової ознаки атаки, що триває, незалежної від конкретного інструмента.

8.5. Платформи Threat Intelligence

MISP (Malware Information Sharing Platform) — поширена платформа з відкритим кодом для збору, зберігання, кореляції та обміну індикаторами компрометації між організаціями та спільнотами. Поряд з відкритими рішеннями на ринку присутні комерційні Threat Intelligence Platform (TIP), а також вбудовані модулі Threat Intelligence у складі самих SIEM-платформ, про що йдеться в підрозділі 8.6.

8.6. Інтеграція Threat Intelligence з SIEM

У **Microsoft Sentinel** (розділ 4) окремий розділ Threat Intelligence дає змогу імпортувати індикатори, вручну, через TAXII-канали або конектори до комерційних ТІ-постачальників, і використовувати їх у аналітичних правилах зіставлення (TI matching analytics), а також збагачувати інциденти автоматично. У **Splunk** (розділ 5) Threat Intelligence framework дає змогу підключати фіди індикаторів (у форматі STIX/TAXII або CSV) і використовувати їх у кореляційних пошуках Enterprise Security для автоматичного збагачення notable events.

Практична цінність інтеграції проста для демонстрації: подія, що сама собою виглядає нейтрально, наприклад з'єднання до зовнішньої IP-адреси, за наявності збігу з відомим індикатором компрометації миттєво підвищує пріоритет розслідування, замикаючи логічне коло всього модуля 1: від сирої події (розділ 2) через кореляцію (розділ 3) до контексту, що робить цю кореляцію по-справжньому цінною.

8.7. Оцінювання якості та достовірності Threat Intelligence

Не всі індикатори, що надходять із джерел підрозділу 8.3, однаково надійні: застарілий IOC чи хибно атрибутований звіт, застосований без критичної оцінки, породжує ті самі хибнопозитивні спрацювання, проблема яких уже неодноразово поставала в цьому модулі (підрозділи 3.4, 7.5). Для формалізованої оцінки джерел і самих даних історично застосовується **Адміралтейська шкала (Admiralty Code)**, що виникла у військовій розвідці й перенесена у кіберрозвідку: джерело оцінюється за літерною шкалою надійності (від А, повністю надійне, до F, надійність не може бути оцінена), а сама інформація, окремо, за числовою шкалою достовірності (від 1, підтверджено іншими джерелами, до 6, достовірність не може бути оцінена). Комбінація, наприклад «B2», дає аналітику стислу, стандартизовану позначку якості кожного фрагмента розвідданих ще до його використання в правилах SIEM.

Окремою вимогою до якісної Threat Intelligence є **строк придатності (TTL, time to live)** індикатора: IP-адреса зловмисної інфраструктури часто змінює власника протягом днів чи тижнів, тому платформа (підрозділ 8.5) має не лише накопичувати індикатори, а й своєчасно позначати застарілі як неактивні, інакше правило зіставлення почне генерувати хибнопозитивні спрацювання на вже легітимний ресурс.

8.8. Threat Hunting: проактивне використання Threat Intelligence

Усі механізми, розглянуті дотепер, від правил кореляції SIEM (розділ 3) до зіставлення індикаторів (підрозділ 8.6), працюють реактивно: система очікує на подію чи збіг і лише тоді сповіщає аналітика. **Threat Hunting (проактивний пошук загроз)** доповнює цю модель протилежним підходом: кваліфікований аналітик самостійно формулює гіпотезу про можливу присутність зловмисника, спираючись саме на тактичний і технічний рівні Threat Intelligence (підрозділ 8.1), наприклад «застосовує метод бічного руху, характерний для угруповання X, який щойно фігурував у новому звіті», і активно перевіряє цю гіпотезу в наявних даних, замість очікування спрацювання правила.

Практично полювання спирається на ту саму матрицю MITRE ATT&CK, згадану в підрозділі 8.4, як спільну мову для формулювання гіпотез («чи є в наших журналах ознаки техніки T1059 за останні 30 днів») та ті самі джерела подій, що вже накопичені в SIEM (журнали з розділу 2, потоки NetFlow з підрозділу 6.4, дані UEBA з підрозділу 3.7). Успішне полювання, що підтвердило гіпотезу, як правило, завершується не лише усуненням конкретної загрози, а й формалізацією нового постійного правила кореляції, яке автоматизує виявлення цього патерну в майбутньому: проактивний пошук у такий спосіб систематично живить і вдосконалює реактивний контур моніторингу, з якого починався розділ 1.

8.9. Атрибуція атак: угруповання загроз та їхні профілі

Підрозділ 8.8 навів приклад гіпотези полювання, що спирається на «угруповання X»; розглянемо, звідки беруться подібні профілі. Дослідники Threat Intelligence, зіставляючи індикатори компрометації (підрозділ 8.4) та тактики й техніки (MITRE ATT&CK) з десятків розслідуваних інцидентів, поступово формують стійкі профілі конкретних **угруповань загроз (threat actor groups)**: набори характерних, повторюваних інструментів, інфраструктури та тактик, що дозволяють з певним рівнем достовірності (шкала Адміралтейства, підрозділ 8.7) приписати нову атаку раніше відомому угрупованню, навіть якщо конкретні використані індикатори (IP-адреси, домени) є новими. Такі профілі отримують умовні позначення (наприклад, за системою АРТ-номерів чи власними назвами постачальників розвідданих), під якими й фігурують у публічних звітах.

Практична цінність атрибуції для SOC-аналітика подвійна: по-перше, зіставлення поточної атаки з відомим угрупованням одразу підказує ймовірні наступні кроки зловмисника, оскільки угруповання, як правило, повторюють свій

характерний ланцюг атаки (kill chain) від кампанії до кампанії; по-друге, атрибуція допомагає оцінити ймовірну мотивацію (фінансова вигода, промислове шпигунство, геополітично мотивована диверсія), що впливає на пріоритизацію реагування. Водночас атрибуція завжди лишається ймовірнісною оцінкою, а не математичним доведенням: досвідчені угруповання свідомо імітують почерк інших, щоб ускладнити чи спотворити атрибуцію (false flag), тому висновок про конкретного виконавця атаки варто формулювати з тим самим рівнем обережності щодо достовірності джерела, що вводився в підрозділі 8.7.

8.10. Обмін розвідданими на національному й галузевому рівні в Україні

Підрозділ 8.3 згадав CERT-UA й галузеві ISAC/ISAO як джерела Threat Intelligence; для українського контексту цей канал має особливе практичне значення, оскільки саме CERT-UA (Урядова команда реагування на комп'ютерні надзвичайні події України) виконує роль національного координаційного центру: приймає повідомлення про інциденти від організацій (обов'язкове для об'єктів критичної інфраструктури згідно з нормативною базою, згаданою в підрозділі 1.3), публікує власні розвіддані про актуальні для України кампанії й угруповання (підрозділ 8.9), і координує обмін індикаторами компрометації між постраждалими організаціями галузі.

Практична організація такого обміну спирається на принцип **довіри в колі учасників**: організація, що ділиться виявленням у себе індикатором компрометації, розраховує на певний рівень конфіденційності щодо джерела цієї інформації (щоб факт компрометації сам по собі не завдав репутаційної шкоди), для чого міжнародна практика виробила протокол маркування **TLP (Traffic Light Protocol)**: кольорова позначка (RED, AMBER, GREEN, CLEAR) при кожному повідомленні прямо вказує одержувачу межі допустимого подальшого поширення отриманої інформації, від «лише для учасників цієї конкретної розмови» (RED) до «можна поширювати вільно» (CLEAR). Дотримання TLP-позначок є передумовою довіри всередині галузевого ISAC і, зрештою, готовності інших організацій ділитися своїми даними в майбутньому.

8.11. MITRE ATT&CK як спільна мова Threat Intelligence і SOC

Матриця **MITRE ATT&CK**, уже згадана побіжно в підрозділах 6.9 і 8.9 як джерело назв конкретних технік, заслуговує на окремий, зведений розгляд, оскільки вона виконує роль спільного словника, що об'єднує розрізнені підрозділи цієї книги

в єдину систему координат. Матриця структурує поведінку зловмисника за **тактиками** (Reconnaissance, Initial Access, Lateral Movement, Exfiltration та інші, всього чотирнадцять на момент написання) і конкретними **техніками** та **підтехніками** в межах кожної тактики, з унікальним ідентифікатором для кожної (наприклад, T1071 для «Application Layer Protocol»).

Практична цінність такої спільної мови проявляється на кожному етапі циклу безпеки, розглянутого в цій книзі: аналітик Threat Intelligence (розділ 8) описує профіль конкретного угруповання через перелік технік АТТ&СК, які воно історично застосовувало; правило кореляції SIEM (підрозділ 3.3) чи аналітичне правило Sentinel (підрозділ 4.5) в описі прямо посилається на ідентифікатор техніки, яку воно покликане виявити; а програма аудиту (розділ 17 модуля 3) може оцінювати покриття організації техніками АТТ&СК як формальний, вимірюваний критерій повноти захисту, а не суб'єктивне враження. Саме завдяки такій наскрізній застосовності АТТ&СК стала фактичним галузевим стандартом де-факто, хоча формально не є ISO- чи NIST-стандартом у строгому сенсі цих слів.

Практикум 8. Імпорт індикаторів компрометації та побудова правила збагачення

Мета: отримати навички роботи з даними Threat Intelligence у SIEM.

Хід роботи:

1. отримати тестовий набір індикаторів компрометації (навчальний фід IOC у форматі CSV/STIX або експорт з інстанції MISP);
2. імпортувати індикатори в обраний SIEM (розділ Threat Intelligence у Sentinel або TI framework у Splunk);
3. написати правило зіставлення (KQL/SPL), що порівнює мережеві журнали з переліком імпортованих індикаторів;
4. згенерувати тестову подію із заздалегідь відомою «шкідливою» IP-адресою зі списку індикаторів, перевірити спрацювання правила;
5. оформити висновок про те, як інтеграція TI змінює пріоритизацію інцидентів порівняно з роботою без неї.

Контрольні питання до розділу 8

1. Назвіть чотири рівні Threat Intelligence та поясніть різницю між ними.
2. Опишіть етапи життєвого циклу кіберрозвідки, зображені на рисунку 8.1.
3. Чим стандарт STIX відрізняється від TAXII?
4. Чим Indicator of Compromise відрізняється від Indicator of Attack?
5. Яку практичну роль виконує матриця MITRE ATT&CK у роботі SOC?
6. Як інтеграція Threat Intelligence із SIEM впливає на пріоритизацію інцидентів?
7. Опишіть різницю між стратегічним, оперативним, тактичним і технічним рівнями Threat Intelligence.
8. З яких шести етапів складається Intelligence Cycle, зображений на рисунку 8.1?
9. Чим комерційне джерело Threat Intelligence відрізняється від відкритого (OSINT)?
10. Що таке TIP (Threat Intelligence Platform) і яку функцію вона виконує?
11. Наведіть приклад того, як формат STIX структурує опис індикатора компрометації.
12. Яку роль виконує протокол TAXII у обміні даними Threat Intelligence між організаціями?
13. Чим тактика (tactic) відрізняється від техніки (technique) у матриці MITRE ATT&CK?

14. Що таке «збагачення» (enrichment) події за допомогою індикаторів компрометації?
15. Наведіть приклад ризику хибного індикатора компрометації (false IOC) і його впливу на роботу SOC.
16. Яким чином автоматизований аналіз телеметрії за допомогою II-агентів (розділ 1) пов'язаний із задачами Threat Intelligence?

МОДУЛЬ 2. АДМІНІСТРУВАННЯ ЗАХИЩЕНИХ ІТ-СИСТЕМ І МЕРЕЖ

Відповідає Змістовому модулю 2 робочої програми: Тема 3 «Адміністрування інформаційних систем і політика безпеки», Тема 4 «Системи виявлення вторгнень».

РОЗДІЛ 9. АДМІНІСТРУВАННЯ ІНФОРМАЦІЙНИХ СИСТЕМ І ПОЛІТИКА БЕЗПЕКИ ІТС



Зміст розділу

- 9.1. Поняття та завдання адміністрування інформаційної системи**
- 9.2. Структура документів політики інформаційної безпеки**
- 9.3. Життєвий цикл адміністрування ІТС**
- 9.4. Ролі, обов'язки та принцип найменших привілеїв**
- 9.5. Управління змінами та документування конфігурацій**
- 9.6. Управління вразливостями (Vulnerability Management)**
- 9.7. Управління активами ІТ та конфігураційна база даних (CMDB)**
- 9.8. Патч-менеджмент: тестування, вікна обслуговування, відкат**
- 9.9. Документація адміністратора та бази знань (runbooks)**
- 9.10. Еталонні базові профілі конфігурації (CIS Benchmarks)**

9.1. Поняття та завдання адміністрування інформаційної системи

Адміністрування інформаційної системи являє собою сукупність організаційних і технічних дій, спрямованих на підтримання працездатності, доступності та захищеності інформаційно-телекомунікаційної системи (ІТС)

протягом усього її життєвого циклу. На відміну від моніторингу, який лише спостерігає, адміністрування активно змінює стан системи: встановлює оновлення, налаштовує служби, створює та видаляє облікові записи, застосовує засоби захисту.

До основних завдань адміністрування захищеної ІТС належать: підтримання актуальності програмного забезпечення та своєчасне усунення вразливостей; керування обліковими записами та правами доступу користувачів; забезпечення безперервності роботи критичних служб; ведення документації щодо конфігурації системи; виконання вимог політики інформаційної безпеки організації на технічному рівні. Адміністратор ІТС є тією ланкою, яка перетворює вимоги політики безпеки, сформульовані керівництвом, на конкретні технічні налаштування.

Управління інформаційною мережею в цьому широкому сенсі, включно з управлінням якістю послуг, білінгом та архітектурою управління на прикладному рівні, детально розкрито в навчальному посібнику з грифом Міністерства освіти і науки України «Управління та якість послуг інформаційних мереж зв'язку», де окремі розділи присвячені національним та міжнародним стандартам управління мережею та засобам віддаленого контролю обладнання для підтримання якості послуг.

9.2. Структура документів політики інформаційної безпеки

Політика інформаційної безпеки організації не існує як єдиний документ, а формує ієрархію документів різного рівня деталізації та юридичної ваги. Рисунок 9.1 показує чотири рівні цієї ієрархії: від стратегічної політики, яку затверджує керівництво, до робочих інструкцій конкретного співробітника.



*Рис. 9.1. Ієрархія документів політики інформаційної безпеки організації
Джерело: розроблено автором*

Політика інформаційної безпеки формулює стратегічні цілі та зобов'язання організації щодо захисту інформації, зазвичай на кількох сторінках, без технічних подробиць. **Стандарти та регламенти** конкретизують обов'язкові технічні вимоги: наприклад, мінімальну довжину пароля, дозволені алгоритми шифрування, перелік дозволеного програмного забезпечення. **Процедури** описують покрокові дії для виконання конкретної задачі, наприклад створення нового облікового запису або реагування на інцидент. **Робочі інструкції та угоди про рівень обслуговування (SLA)** діють на рівні окремого співробітника, робочого місця чи конкретної системи.

Практична причина, з якої важливо розрізняти ці рівні: зміна стандарту (наприклад, вимоги до складності пароля) не повинна вимагати перегляду всієї політики, а лише відповідного документа нижчого рівня. Це дозволяє організації швидко адаптуватися до нових загроз, не порушуючи стабільності стратегічних документів.

9.3. Життєвий цикл адміністрування ІТС

Адміністрування захищеної ІТС є не одноразовою дією, а безперервним циклом, який в управлінні якістю прийнято описувати моделлю PDCA (Plan-Do-Check-Act, Plan-Do-Check-Act). Рисунок 9.2 адаптує цю модель до задач адміністратора ІТС.



Рис. 9.2. Цикл адміністрування ІТ-системи (адаптована модель PDCA)

Джерело: розроблено автором

На етапі **планування** адміністратор аналізує поточний стан системи, визначає цілі змін і оцінює ризики. На етапі **впровадження** зміни застосовуються: встановлюються оновлення, змінюються налаштування, розгортаються нові служби. Етап **перевірки** підтверджує, що зміна досягла мети і не спричинила побічних ефектів, часто із залученням даних моніторингу з розділів 1-8. На етапі **коригування** результати перевірки або закріплюються як новий стандартний стан системи, або зміна відкочується, якщо перевірка виявила проблему. Цикл повторюється безперервно, оскільки жодна ІТС не є статичною: з'являються нові вразливості, оновлюється апаратне й програмне забезпечення, змінюються бізнес-вимоги.

9.4. Ролі, обов'язки та принцип найменших привілеїв

Адміністрування великої ІТС рідко виконується однією людиною з необмеженими правами. Типовий розподіл ролей включає адміністратора мережі (відповідає за маршрутизатори, комутатори, канали зв'язку), адміністратора серверів (операційні системи, служби), адміністратора баз даних, адміністратора безпеки (розділ 10) та адміністратора хмарної інфраструктури (розділ 14). У невеликих організаціях ці ролі можуть поєднуватися в однієї людини, але навіть тоді доцільно розділяти рівні доступу за призначенням облікового запису.

Ключовим принципом розподілу прав є **принцип найменших привілеїв (Principle of Least Privilege, PoLP)**: кожен обліковий запис, процес чи система повинні мати мінімальний обсяг прав, необхідний для виконання своїх функцій, і не більше. Практичним наслідком цього принципу є відмова від щоденної роботи під обліковим записом з правами адміністратора: навіть системний адміністратор повинен мати окремий, непривілейований обліковий запис для повсякденних задач і використовувати підвищення прав (*sudo*, *Run as Administrator*, *PIM*) лише для конкретних адміністративних дій, з фіксацією факту підвищення прав у журналі аудиту.

9.5. Управління змінами та документування конфігурацій

Незадокументована зміна конфігурації є одним із найпоширеніших джерел інцидентів доступності: адміністратор змінює правило міжмережевого екрана для тестування і забуває повернути його; налаштування, зроблене шість місяців тому, ніхто вже не пам'ятає й не може пояснити. Процес **управління змінами (Change Management)** формалізує внесення змін до продуктивної системи: кожна зміна проходить етапи запиту, оцінки впливу й ризиків, затвердження відповідальною особою, планового впровадження (за можливості в регламентоване вікно обслуговування) і фіксації в реєстрі змін.

Невіддільною частиною управління змінами є підтримання актуальної документації конфігурації: схем мережі, переліку встановленого програмного забезпечення й версій, списку відкритих портів і правил доступу. У розвинених організаціях цю документацію дедалі частіше замінює або доповнює підхід **Infrastructure as Code**, коли сама конфігурація системи описується у вигляді файлів, що зберігаються в системі контролю версій; детально ця концепція розглядається в розділі 14 стосовно хмарної інфраструктури.

9.6. Управління вразливостями (Vulnerability Management)

Якщо принцип найменших привілеїв (підрозділ 9.4) обмежує наслідки успішної атаки, то управління вразливостями спрямоване на те, щоб зменшити саму кількість можливих шляхів для такої атаки. Процес будується як регулярний цикл: **сканування** інфраструктури спеціалізованим засобом (наприклад Nessus, OpenVAS) для виявлення відомих слабких місць у програмному забезпеченні й конфігурації; **класифікація** кожної знайденої вразливості за загальновизнаним ідентифікатором **CVE (Common Vulnerabilities and Exposures)**, що дає змогу однозначно посилатися на конкретну вразливість незалежно від постачальника засобу сканування; **оцінювання критичності** за шкалою **CVSS (Common Vulnerability Scoring System)**, числовим показником від 0 до 10, що враховує вектор атаки, складність експлуатації та необхідний рівень привілеїв, згадану раніше в контексті практикуму з класифікації результатів сканування в розділі 1 модуля 1; і, нарешті, **усунення** через встановлення оновлення (патч-менеджмент) або компенсаційний контроль, якщо офіційний патч ще не випущено.

Практична пріоритизація усунення не повинна спиратися лише на числовий бал CVSS: вразливість із високим балом на ізольованому тестовому сервері становить менший практичний ризик, ніж вразливість із середнім балом на загальнодоступному вебсервері з відомим публічним експлойтом. Тому зрілі процеси управління вразливостями враховують додатково критичність активу (підрозділ 9.7) і наявність підтвердженого експлойту в дикій природі.

9.7. Управління активами ІТ та конфігураційна база даних (CMDB)

Ефективне сканування вразливостей, описане в підрозділі 9.6, і обґрунтоване оцінювання впливу зміни, описане в підрозділі 9.5, однаково спираються на попередню умову: організація повинна достеменно знати, які активи (сервери, робочі станції, мережеве обладнання, віртуальні машини, хмарні ресурси) взагалі існують у її інфраструктурі. Систему не можна захистити чи просканувати, якщо адміністратор не знає про її існування, тому провадиться безперервна **інвентаризація активів (IT Asset Management)**: активне сканування мережі для виявлення нових вузлів, агенти інвентаризації на кінцевих станціях, інтеграція з API хмарного провайдера для автоматичного виявлення ресурсів, створених поза стандартною процедурою (розділ 14).

Результати інвентаризації типово накопичуються в **конфігураційній базі даних (Configuration Management Database, CMDB)**: центральному репозиторії

конфігураційних одиниць (Configuration Item, CI) та зв'язків між ними, наприклад «застосунок X виконується на сервері Y, який залежить від бази даних Z». Саме ці зв'язки роблять CMDB цінною для оцінювання впливу зміни на етапі планування циклу PDCA (рисунок 9.2): адміністратор бачить наперед, які ще системи постраждають від зміни конкретного компонента, замість того щоб з'ясовувати це постфактум через інцидент.

9.8. Патч-менеджмент: тестування, вікна обслуговування, відкат

Підрозділ 9.6 назвав встановлення оновлення завершальним етапом циклу управління вразливостями, але саме цей, здавалося б, найпростіший крок найчастіше спричиняє інциденти доступності, якщо виконується необдуманно: оновлення, що усуває одну вразливість, теоретично може одночасно порушити сумісність із наявним застосунком чи спричинити непередбачену відмову служби. Тому зрілий **патч-менеджмент** ніколи не зводиться до негайного встановлення кожного випущеного оновлення на всі системи одночасно, а будується як окремий, керований процес.

Типовий процес передбачає: класифікацію оновлень за критичністю (виправлення активно експлуатованої вразливості вимагає термінового встановлення, тоді як планове оновлення функціональності може почекати регулярного циклу); попереднє тестування оновлення в ізольованому тестовому середовищі, що є фізичним чи віртуальним відображенням продуктивної конфігурації; поетапне розгортання за завчасно узгодженим **вікном обслуговування (maintenance window)**, спершу на невеликій контрольній групі систем, а вже потім на решті інфраструктури; і обов'язково підготовлений **план відкату (rollback plan)** на випадок, якщо оновлення все ж спричинить непередбачену проблему вже після встановлення. Останній пункт прямо спирається на резервне копіювання конфігурації, розглянуте в підрозділі 11.5, без якого швидкий і безпечний відкат технічно неможливий.

9.9. Документація адміністратора та бази знань (runbooks)

Управління змінами (підрозділ 9.5) вимагає документування конфігурації системи, але окремою, часто недооціненою категорією документації є **процедурна документація**: покроковий опис того, як виконати конкретну регулярну чи аварійну операцію, доступний не лише тому адміністратору, що її розробив, а будь-кому в команді. **Runbook** (експлуатаційний посібник) описує стандартну, регулярну

процедуру (наприклад, «як додати новий сегмент мережі до системи моніторингу з розділів 1-8»), тоді як **playbook реагування на інцидент**, вужчий за призначенням документ, описує послідовність дій для конкретного типу нештатної ситуації (детальніше розглянуто разом з Incident Response Plan у підрозділі 10.5).

Практична цінність такої документації розкривається саме в момент, коли єдиний співробітник, що «тримав усе в голові», недоступний: у відпустці, звільнився, чи сам є об'єктом розслідування інциденту. Організація, що покладається на неформальні знання конкретних людей замість формалізованих runbook, стикається з тим самим ризиком, що вже згадувався в підрозділі 9.5 стосовно недокументованих змін конфігурації, тільки в масштабі цілих процедур, а не окремих налаштувань. Дедалі частіше такі процедури не просто описуються текстом, а автоматизуються безпосередньо у вигляді сценаріїв (скриптів чи Ansible-плейбуків, згаданих у підрозділі 11.4), що перетворює документацію із суто довідкового матеріалу на виконуваний код. Цей напрям, коли політики безпеки й процедури реагування самі оформлюються як версійований, тестований код (**Policy-as-Code**), а не як окремий текстовий документ, докладніше розглянуто в дослідженні Rudin, Vasylenko, Grynkevych, присвяченому інтеграції DevSecOps-автоматизації, ML-детекції загроз і Policy-as-Code в єдину хмарну платформу безпеки.

9.10. Еталонні базові профілі конфігурації (CIS Benchmarks)

Стандарт безпеки, розглянутий у підрозділі 9.2 як другий рівень ієрархії документів організації, зазвичай формулює вимогу на абстрактному рівні («операційна система повинна бути захищена від несанкціонованого доступу»), не деталізуючи конкретне значення кожного окремого параметра. **CIS Benchmarks**, підготовлені організацією Center for Internet Security, заповнюють саме цю прогалину: для кожної поширеної операційної системи, бази даних чи хмарної платформи публікується документ із сотнями конкретних, перевірюваних налаштувань (наприклад, точна мінімальна довжина пароля, перелік вимкнених за замовчуванням непотрібних служб, конкретні параметри реєстру чи файлу конфігурації), згрупованих за рівнями суворості (Level 1 для базового захисту, Level 2 для середовищ з підвищеними вимогами).

Практична цінність еталонного профілю для адміністратора полягає в переході від суб'єктивного «начебто захищено» до об'єктивно перевірюваного стану: спеціалізовані інструменти сканування (розділ 20 модуля 3) можуть автоматично зіставити фактичну конфігурацію системи з конкретним CIS Benchmark

і сформувати точний перелік відхилень, що технічно є тим самим комплаєнс-скануванням, розглянутим поглиблено в підрозділі 20.7. Використання визнаного галузевого еталона замість власного, розробленого організацією самостійно переліку налаштувань, також суттєво спрощує аргументацію перед зовнішнім аудитором (розділ 18 модуля 3), оскільки CIS Benchmark сам по собі є визнаним, задокументованим джерелом критеріїв аудиту.

Практикум 9. Складання фрагмента процедури управління змінами

Мета роботи: застосувати модель PDCA та ієрархію документів політики безпеки (підрозділи 9.2-9.3) для формалізації конкретної адміністративної зміни.

Хід роботи:

1. обрати умовну зміну конфігурації (наприклад, впровадження нової вимоги до складності пароля або відкриття мережевого порту для нового сервісу) і визначити, до якого рівня ієрархії документів з рисунка 9.1 вона належить;
2. описати цю зміну як процедуру (третій рівень ієрархії): мета, ініціатор, необхідні узгодження, критерії успіху;
3. розписати проходження зміни через чотири етапи циклу PDCA з рисунка 9.2, зазначивши конкретні дії адміністратора на кожному етапі;
4. визначити, які записи повинні потрапити до реєстру змін організації після завершення впровадження;
5. оформити результат як внутрішній документ управління змінами із зазначенням відповідальної особи та дати перегляду.

Контрольні питання до розділу 9

1. Чим адміністрування ІТС принципово відрізняється від моніторингу та аудиту?
2. Опишіть чотири рівні ієрархії документів політики інформаційної безпеки.
3. Яку мету переслідує кожен з чотирьох етапів циклу PDCA стосовно адміністрування ІТС?
4. У чому полягає принцип найменших привілеїв і чому підвищення прав повинно фіксуватися окремо?
5. Які ризики створює незадокументована зміна конфігурації продуктивної системи?
6. Наведіть приклад стандарту (другого рівня ієрархії з рисунка 9.1), що конкретизує вимогу політики безпеки.
7. Чому зміна стандарту не повинна вимагати перегляду всієї політики інформаційної безпеки?
8. Опишіть, що відбувається на кожному з чотирьох етапів циклу PDCA (рисунок 9.2) стосовно конкретної зміни конфігурації.
9. Дайте означення принципу найменших привілеїв (PoLP) і наведіть практичний приклад його порушення.

10. Чому системний адміністратор повинен мати окремий непривілейований обліковий запис для повсякденних задач?
11. Що таке підвищення прав (sudo, Run as Administrator, PIM) і чому воно має фіксуватися в журналі аудиту?
12. Перелічіть типові ролі адміністративного персоналу великої ІТС.
13. Що таке процес управління змінами (Change Management) і з яких етапів він складається?
14. Які ризики створює незадокументована зміна конфігурації продуктивної системи для безперервності бізнесу?
15. Чим підхід Infrastructure as Code (розділ 14) є логічним продовженням практики документування конфігурації?
16. Наведіть приклад робочої інструкції чи угоди SLA (четвертий рівень ієрархії політики) для конкретної системи.

РОЗДІЛ 10. ЗАХИСТ ІНФОРМАЦІЇ В КОМП'ЮТЕРНИХ МЕРЕЖАХ. ФУНКЦІЇ АДМІНІСТРАТОРА БЕЗПЕКИ



Зміст розділу

- 10.1. Функції та повноваження адміністратора безпеки**
- 10.2. Багаторівневий захист мережі (Defense in Depth)**
- 10.3. Керування доступом: автентифікація, авторизація, аудит (AAA)**
- 10.4. Сегментація мережі та міжмережеві екрани**
- 10.5. Роль адміністратора безпеки в реагуванні на інциденти**
- 10.6. Віддалений доступ: VPN та засади Zero Trust Network Access**
- 10.7. Управління привілейованим доступом (Privileged Access Management, PAM)**
- 10.8. Аудит і оптимізація правил міжмережевого екрана**
- 10.9. Інфраструктура відкритих ключів (PKI) і керування сертифікатами**

10.10. Контроль доступу до мережі на каналному рівні (НАС, 802.1X)

10.1. Функції та повноваження адміністратора безпеки

Адміністратор безпеки посідає окрему позицію серед адміністративного персоналу ІТС: його завданням є не забезпечення функціональності системи як такої, а контроль за дотриманням вимог захищеності всіма іншими компонентами й адміністраторами. До типових функцій адміністратора безпеки належать: впровадження й контроль дотримання політики безпеки на технічному рівні; керування правами доступу користувачів і періодичний перегляд (рецертифікація) наданих прав; налаштування та супровід засобів захисту (міжмережевих екранів, систем виявлення вторгнень, антивірусних рішень); аналіз журналів подій безпеки спільно із засобами, розглянутими в розділах 2-3; координація реагування на інциденти інформаційної безпеки.

Важливою організаційною вимогою є розділення обов'язків адміністратора безпеки й адміністратора системи: особа, що надає права доступу, за можливості не повинна бути тією самою особою, що перевіряє коректність цього надання. У невеликих організаціях, де повне розділення неможливе, компенсаційним заходом є обов'язковий незалежний аудит дій адміністратора (модуль 3).

10.2. Багаторівневий захист мережі (Defense in Depth)

Жоден окремий засіб захисту, яким би досконалим він не був, не гарантує захищеності мережі: міжмережевий екран можна обійти через сконфігурований виняток, антивірус не розпізнає нову модифікацію шкідливого коду. Тому в основу побудови захищеної мережі покладено принцип **ешелонованого захисту (Defense in Depth)**: декілька незалежних рубежів захисту, розташованих так, що подолання одного з них не означає компрометації всієї системи. Рисунок 10.1 зображує цей принцип у вигляді концентричних рубежів навколо об'єкта захисту.

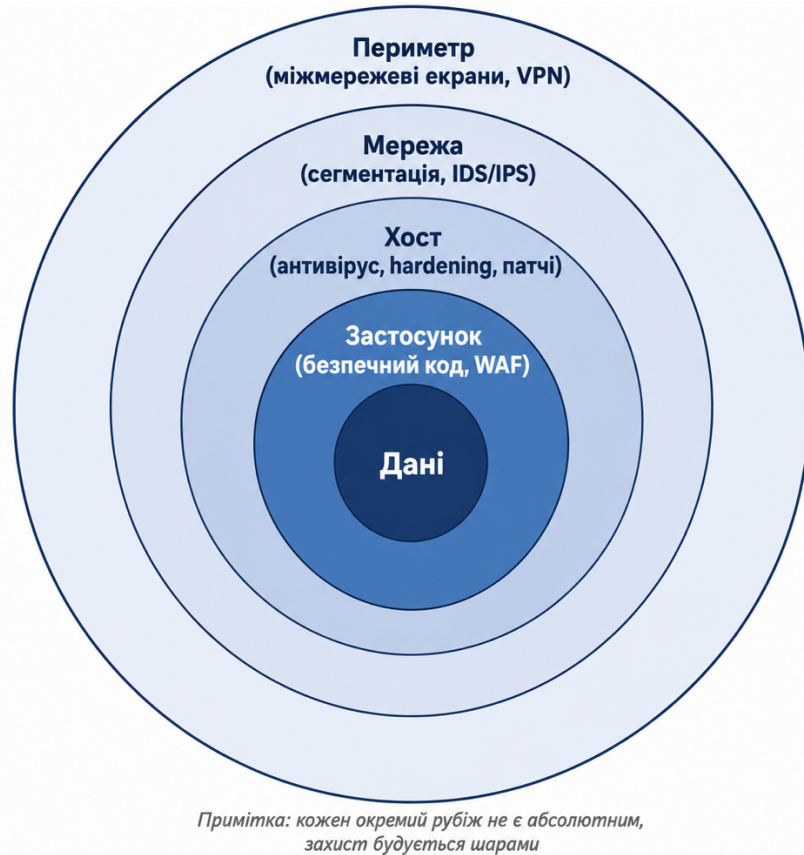


Рис. 10.1. Багаторівневий захист мережі за принципом *Defense in Depth*
Джерело: розроблено автором

Кожен рубіж на рисунку 10.1 відповідає за свою частину загроз: периметр (міжмережевий екран, VPN-шлюз) відсіює більшість зовнішнього небажаного трафіку; сегментація мережі та системи виявлення вторгнень (розділ 12) обмежують поширення атаки всередині мережі, якщо периметр подолано; захист хоста (оновлення, антивірус, hardening операційної системи) протидіє загрозам, що дісталися конкретного вузла; захист застосунку (безпечна розробка, WAF) захищає найближчий до даних рівень. Практичний висновок для адміністратора безпеки: інвестиції лише в найдорожчий периметровий міжмережевий екран за нехтування рештою рубежів створюють хибне відчуття захищеності.

10.3. Керування доступом: автентифікація, авторизація, аудит (AAA)

Керування мережевим доступом традиційно описують моделлю AAA: **автентифікація (Authentication)** підтверджує, що користувач або пристрій є тим, за кого себе видає; **авторизація (Authorization)** визначає, які дії підтвердженому суб'єкту дозволено виконувати; **аудит (Accounting)** фіксує, які дії фактично було

виконано. Для мережевого обладнання (комутатори, VPN-концентратори, безпроводові точки доступу) цю модель зазвичай реалізують централізовано за допомогою протоколів RADIUS або TACACS+, порівняних у таблиці 10.1.

Таблиця 10.1. Порівняння протоколів централізованої автентифікації RADIUS і TACACS+

Характеристика	RADIUS	TACACS+
Транспортний протокол	UDP	TCP
Шифрування пакета	лише поле пароля	весь пакет
Розділення автентифікації й авторизації	об'єднані в одному обміні	розділені окремі процеси
Типове застосування	безпроводовий доступ, VPN, каналний рівень (802.1X)	адміністративний доступ до мережевого обладнання

Вибір протоколу на практиці залежить від задачі: для контролю доступу користувачів до мережі (802.1X) типово застосовують RADIUS, тоді як для централізованого контролю адміністративного доступу до самого мережевого обладнання, з докладним журналюванням кожної виконаної команди, перевагу традиційно надають TACACS+.

10.4. Сегментація мережі та міжмережеві екрани

Сегментація мережі, поділ єдиної мережі на ізольовані одна від одної логічні частини (VLAN, підмережі, мікросегменти), є одним із найефективніших і водночас найдешевших заходів обмеження наслідків компрометації. Мережа, поділена на сегменти за призначенням (робочі станції користувачів, сервери, гостьовий доступ, технологічне обладнання), не дозволяє зловмиснику, що скомпрометував один робочий вузол, безперешкодно просуватися мережею до критичних серверів.

Міжмережевий екран (firewall) контролює трафік між сегментами мережі за встановленими правилами. Сучасні рішення класу NGFW (Next-Generation Firewall) поєднують традиційну фільтрацію за адресами й портами з глибокою інспекцією пакетів, розпізнаванням застосунків незалежно від використовуваного порту та інтеграцією з базами репутації загроз. Адміністратор безпеки відповідає за підтримання правил міжмережевого екрана в актуальному стані: періодичний

перегляд правил, видалення застарілих винятків і документування причини кожного нестандартного дозволу є звичайною практикою, що суттєво знижує поверхню атаки.

10.5. Роль адміністратора безпеки в реагуванні на інциденти

Коли моніторинг (розділи 1-8) фіксує ознаки інциденту, саме адміністратор безпеки, як правило, очолює або координує технічну частину реагування: ізоляцію скомпрометованого вузла від мережі, тимчасове блокування облікового запису, аналіз масштабу компрометації, взаємодію із зовнішніми структурами реагування (в Україні, зокрема, з CERT-UA у випадках, що підпадають під законодавчо визначені критерії). Формалізований план реагування на інциденти (Incident Response Plan), підготовлений заздалегідь, а не імпровізований під час самого інциденту, є ознакою зрілої практики адміністрування безпеки; докладніше процес аудиту наслідків інциденту розглянуто в модулі 3.

10.6. Віддалений доступ: VPN та засади Zero Trust Network Access

Периметр на рисунку 10.1 давно перестав бути суцільним фізичним кордоном: співробітники підключаються до корпоративних ресурсів з домашніх мереж, готелів, публічних Wi-Fi-точок. Традиційним рішенням є **VPN (Virtual Private Network)**: шифрований тунель між віддаленим вузлом і корпоративною мережею, що розрізняють за призначенням на **site-to-site VPN** (постійне з'єднання між двома майданчиками організації, типово на основі протоколу IPsec) і **remote access VPN** (з'єднання окремого користувача з мережею організації, дедалі частіше на основі SSL/TLS, що не вимагає встановлення спеціалізованого клієнта складної конфігурації).

Класична модель remote access VPN має суттєве обмеження з погляду безпеки: після успішного підключення користувач зазвичай отримує широкий мережевий доступ до всього внутрішнього сегмента, що суперечить принципу найменших привілеїв (підрозділ 9.4). Модель **Zero Trust Network Access (ZTNA)**, що дедалі активніше витісняє класичний VPN у нових архітектурах, усуває це припущення: жодному підключенню не довіряють апріорно лише на підставі його джерела чи факту перебування «всередині» периметра; кожен запит доступу до конкретного застосунку перевіряється окремо, з урахуванням особистості користувача, стану захищеності його пристрою й контексту запиту, а доступ надається до конкретного застосунку, а не до всього мережевого сегмента. Принцип «ніколи не довіряй,

завжди перевіряй» (never trust, always verify), покладений в основу ZTNA, є логічним продовженням ешелонованого захисту з підрозділу 10.2, перенесеним із рівня мережевого периметра на рівень кожного окремого запиту доступу.

10.7. Управління привілейованим доступом (Privileged Access Management, PAM)

Принцип найменших привілеїв (підрозділ 9.4) для звичайного користувача реалізується відносно просто, однак адміністративні (привілейовані) облікові записи, здатні змінювати конфігурацію систем, становлять непропорційно високий ризик у разі компрометації: зловмисник, що заволодів одним таким обліковим записом, найчастіше отримує можливість безперешкодно рухатися мережею. Клас рішень **PAM (Privileged Access Management)** спеціалізується саме на контролі цієї категорії доступу: паролі привілейованих облікових записів зберігаються в централізованому захищеному сховищі (vault) і не відомі адміністратору в незмінному вигляді, а видаються тимчасово під конкретне завдання (just-in-time access) із автоматичною ротацією після завершення сеансу; кожна привілейована сесія записується для подальшого аудиту, аналогічно до журналювання підвищення прав, згаданого в підрозділі 9.4.

Впровадження PAM зазвичай починають з інвентаризації всіх привілейованих облікових записів організації, включно із «сервісними» обліковими записами застосунків (які часто взагалі випадають з уваги адміністраторів, оскільки не належать конкретній людині), спираючись на ту саму базу активів CMDB, що згадувалася в підрозділі 9.7.

10.8. Аудит і оптимізація правил міжмережевого екрана

Підрозділ 10.4 назвав періодичний перегляд правил обов'язковою практикою адміністратора безпеки; розглянемо, чому набір правил міжмережевого екрана взагалі схильний до неконтрольованого розростання й що з цим робити. З часом до правил додаються тимчасові винятки для тестування, які ніхто не прибирає після завершення тесту; правила, що дублюють одне одного частково або повністю; правила для систем, які вже виведені з експлуатації, але формально лишаються дозволеними. Кожне таке «мертве» правило не лише захаращує конфігурацію, а й потенційно відкриває шлях, яким адміністратор давно не переймається, чим охоче користується зловмисник, що проводить розвідку периметра.

Формальний **аудит правил** передбачає регулярний (наприклад, щоквартальний) перегляд повного набору правил із залученням власників

відповідних систем: для кожного правила перевіряється, чи досі актуальна бізнес-причина його існування (документована при створенні, згідно з підрозділом 10.4), чи не перекривається воно ширшим за обсягом правилом, і чи відповідає принципу найменших привілеїв (підрозділ 9.4) в частині дозволених портів і напрямків трафіку. Автоматизовані засоби аналізу правил міжмережевого екрана здатні виявляти явні дублікати й «тіньові» правила (shadowed rules, коли правило ніколи фактично не спрацьовує, бо розташоване нижче за пріоритетом ширшого правила), суттєво прискорюючи цей процес порівняно з ручним переглядом сотень чи тисяч записів.

10.9. Інфраструктура відкритих ключів (PKI) і керування сертифікатами

Практично кожен механізм захисту, розглянутий у цьому розділі й подальших, TLS (підрозділ 13.5), DNSSEC і DKIM (підрозділи 11.2 і 11.6), взаємна автентифікація сервісів (підрозділ 15.7), спирається на криптографію з відкритим ключем, а конкретніше, на довіру до того, що відкритий ключ справді належить заявленому власнику. Цю довіру забезпечує **інфраструктура відкритих ключів (Public Key Infrastructure, PKI)**: ієрархія центрів сертифікації (Certificate Authority, CA), кожен з яких підписує (засвідчує) сертифікати нижчого рівня своїм закритим ключем, утворюючи ланцюг довіри (chain of trust) від кореневого центру сертифікації, якому апріорно довіряють операційні системи й браузері, до сертифіката конкретного сервера чи служби.

Для внутрішніх потреб організації (автентифікація внутрішніх серверів, служб, іноді й співробітників) типово розгортають власний, приватний центр сертифікації, інтегрований зі службою каталогів (підрозділ 11.1), а не покладаються виключно на публічні центри сертифікації, призначені передусім для загальнодоступних вебресурсів. Окремою, критично важливою функцією адміністрування PKI є **відкликання сертифікатів (revocation)**: якщо закритий ключ, що відповідає сертифікату, скомпрометовано (наприклад, при компрометації сервера), сертифікат необхідно негайно позначити недійсним через список відкликаних сертифікатів (CRL) або протокол оперативної перевірки статусу (OCSP), інакше зломисник і надалі зможе видавати себе за легітимний сервіс попри те, що сертифікат формально скомпрометований.

10.10. Контроль доступу до мережі на каналному рівні (NAC, 802.1X)

Сегментація мережі (підрозділ 10.4) визначає, куди дозволено переміщатися трафіку вже підключеного пристрою, але не вирішує окремого, попереднього питання: чи повинен цей конкретний пристрій взагалі отримати фізичне підключення до мережевого порту. **Мережевий контроль доступу (Network Access Control, NAC)** відповідає саме за цей, найраніший рубіж: перш ніж дозволити пристрою обмін трафіком, NAC-рішення перевіряє його особу (через сертифікат чи облікові дані) і, часто, стан захищеності (наявність актуальних оновлень, увімкнений антивірус), і лише після успішної перевірки надає доступ до відповідного VLAN (підрозділ 13.8).

Технічною основою для дротових мереж є протокол **802.1X**, уже згаданий у контексті бездротових мереж підприємства в підрозділі 13.7: комутатор виконує роль автентифікатора, що не пропускає жодного трафіку через порт до успішного проходження автентифікації пристроєм через сервер RADIUS (підрозділ 10.3). Пристрій, що не пройшов автентифікацію (наприклад, особистий ноутбук відвідувача, підключений до вільної розетки в переговорній кімнаті), NAC типово поміщає в ізольований **гостьовий VLAN** з обмеженим доступом лише до інтернету, замість повної відмови в підключенні, що поєднує безпеку з практичною зручністю для легітимних гостьових сценаріїв.

Практикум 10. Розроблення фрагмента політики паролів та мережевої сегментації

Мета роботи: набути практичних навичок формалізації вимог політики безпеки у вигляді технічного документа, придатного для подальшої реалізації засобами адміністрування.

Хід роботи:

1. розробити фрагмент стандарту (другий рівень ієрархії з рисунка 9.1), що визначає вимоги до складності, терміну дії та історії паролів для трьох категорій облікових записів: звичайний користувач, адміністратор, службовий обліковий запис;
2. для навчального сегмента мережі (10-15 вузлів умовної організації) розробити схему сегментації на щонайменше чотири VLAN за призначенням;
3. сформулювати для кожної пари сегментів обґрунтування дозволеного або забороненого трафіку у вигляді таблиці правил міжмережевого екрана;
4. оформити результат як внутрішній технічний документ із зазначенням дати перегляду.

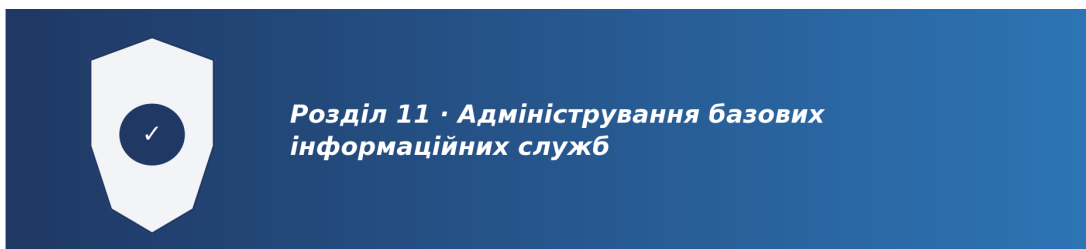
Примітка для викладача. Конкретні числові вимоги (довжина пароля, термін дії) варто звіряти з чинною редакцією стандартів організації-замовника або з NIST SP 800-63B, якщо завдання виконується без прив'язки до конкретної організації.

Контрольні питання до розділу 10

1. Чим обов'язки адміністратора безпеки відрізняються від обов'язків адміністратора системи чи мережі і чому бажане їх розділення?
2. Поясніть принцип Defense in Depth на прикладі мережі з рисунка 10.1.
3. У чому полягає різниця між автентифікацією, авторизацією та аудитом у моделі AAA?
4. Порівняйте протоколи RADIUS і TACACS+ за транспортом, шифруванням та типовим застосуванням.
5. Яку роль відіграє сегментація мережі в обмеженні наслідків компрометації одного вузла?
6. Перелічіть основні функції адміністратора безпеки в організації.
7. Чому бажане розділення обов'язків між адміністратором, що надає права доступу, і особою, що перевіряє коректність цього надання?

8. Опишіть, за яку частину загроз відповідає кожен з чотирьох рубежів на рисунку 10.1.
9. Наведіть приклад ситуації, коли інвестиції лише в периметровий міжмережевий екран створюють хибне відчуття захищеності.
10. Дайте означення кожного з трьох компонентів моделі AAA.
11. У яких сценаріях типово застосовують RADIUS, а в яких TACACS+ (таблиця 10.1)?
12. Що таке сегментація мережі і як вона обмежує наслідки компрометації одного вузла?
13. Чим міжмережевий екран класу NGFW відрізняється від традиційного пакетного фільтра?
14. Яку практику варто застосовувати адміністратору безпеки для підтримання правил міжмережевого екрана в актуальному стані?
15. Яку роль відіграє адміністратор безпеки під час реагування на інцидент?
16. Що таке формалізований план реагування на інциденти (Incident Response Plan) і чому його варто готувати заздалегідь?

РОЗДІЛ 11. АДМІНІСТРУВАННЯ БАЗОВИХ ІНФОРМАЦІЙНИХ СЛУЖБ



Зміст розділу

- 11.1. Служба каталогів та контролер домену***
- 11.2. Служба DNS: адміністрування та захист***
- 11.3. Служба DHCP: адміністрування та типові атаки***
- 11.4. Групові політики та централізоване управління конфігурацією***
- 11.5. Резервне копіювання та відновлення служб***
- 11.6. Автентифікація домену електронної пошти: SPF, DKIM, DMARC***
- 11.7. Синхронізація часу: служба NTP та її роль у безпеці***

11.8. Веб-проксі та фільтрація вихідного вебтрафіку

11.9. Файлові служби та служби друку: адміністрування спільних ресурсів

11.10. Основи захисту серверів баз даних

11.1. Служба каталогів та контролер домену

Більшість корпоративних ІТС спираються на централізовану **службу каталогів**, яка зберігає інформацію про користувачів, комп'ютери, групи та ресурси мережі в єдиній ієрархічній базі. Найпоширенішою реалізацією лишається Active Directory на основі протоколу LDAP, хоча базові принципи однаково застосовні до вільних альтернатив (FreeIPA, OpenLDAP). Рисунок 11.1 показує, як контролер домену взаємодіє із суміжними базовими службами.

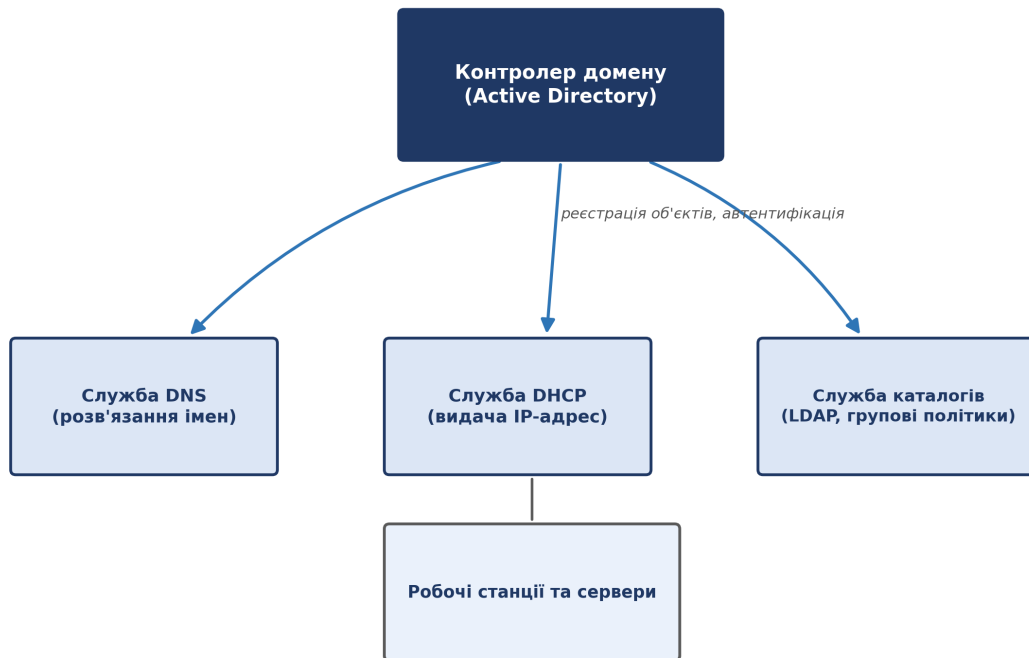


Рис. 11.1. Взаємодія базових інформаційних служб навколо контролера домену

Джерело: розроблено автором

Контролер домену виконує автентифікацію користувачів і комп'ютерів (типово за протоколом Kerberos), зберігає облікові записи та групи, а також групові політики (підрозділ 11.4). З точки зору захищеності контролер домену є одним із найкритичніших вузлів мережі: компрометація облікового запису з правами

адміністратора домену фактично означає компрометацію всієї інфраструктури, що спирається на цей каталог. Тому адміністрування контролерів домену традиційно виділяють в окрему, суворо обмежену адміністративну зону (tier-модель адміністрування, де облікові записи, що керують контролером домену, не використовуються для входу на робочі станції користувачів).

11.2. Служба DNS: адміністрування та захист

Служба **DNS (Domain Name System)** розв'язує символічні імена вузлів мережі в IP-адреси і навпаки. У внутрішній мережі організації DNS зазвичай інтегрована зі службою каталогів: контролери домену одночасно виконують роль DNS-серверів, автоматично реєструючи записи приєднаних до домену вузлів. Адміністрування DNS охоплює підтримання актуальності записів, розмежування зон (внутрішня, зовнішня, DMZ) і контроль за дозволеними джерелами оновлень зони, щоб уникнути несанкціонованої підміни записів.

Захищеність DNS забезпечується розширенням **DNSSEC**, яке додає до відповідей DNS криптографічний підпис, що дозволяє клієнту перевірити автентичність та цілісність отриманої відповіді і протидіяти підміні DNS-відповідей (DNS spoofing, cache poisoning). Для зовнішньої DNS-інфраструктури організації, особливо для доменів, що використовуються для електронної пошти, впровадження DNSSEC разом із записами SPF, DKIM і DMARC є визнаною практикою зниження ризику фішингових атак, що маскуються під легітимний домен організації.

11.3. Служба DHCP: адміністрування та типові атаки

Служба **DHCP (Dynamic Host Configuration Protocol)** автоматично видає вузлам мережі IP-адреси та супутні параметри (маску підмережі, шлюз за замовчуванням, адреси DNS-серверів), звільняючи адміністратора від ручного налаштування кожного вузла. Адміністрування DHCP включає визначення діапазонів (scope) адрес для кожного сегмента мережі, термінів оренди адреси (lease time) та резервування конкретних адрес за критичними пристроями (принтери, сервери керування).

Оскільки DHCP-запит за визначенням надсилається до того, як вузол отримає адресу, і не передбачає автентифікації клієнта, служба вразлива до кількох типових атак: **DHCP starvation**, вичерпання зловмисником усього пулу вільних адрес фіктивними запитами, що унеможливорює отримання адреси легітимними вузлами; **rogue DHCP server**, розгортання зловмисником власного DHCP-сервера в мережі,

який видає підконтрольні йому параметри (зокрема, підмінений шлюз чи DNS-сервер) і таким чином здійснює перехоплення трафіку. Основним технічним заходом протидії є функція **DHCP snooping** на керованих комутаторах, яка дозволяє відповідати на DHCP-запити лише з явно довірених портів.

11.4. Групові політики та централізоване управління конфігурацією

Ручне налаштування кожної робочої станції в організації із сотнями чи тисячами вузлів практично неможливе й неминуче призводить до розбіжностей конфігурації. **Групові політики (Group Policy)** та їхні аналоги в інших екосистемах (Ansible, Puppet, Chef, Microsoft Intune для хмарного управління пристроями) дозволяють централізовано визначити й примусово застосувати налаштування безпеки: параметри пароля, дозволені й заборонені застосунки, налаштування міжмережевого екрана вузла, параметри аудиту.

З точки зору захищеності централізоване управління конфігурацією є не лише інструментом зручності, а прямим засобом контролю за виконанням стандартів, розглянутих у розділі 9: замість сподівання, що кожен користувач самостійно дотримається вимог, адміністратор технічно унеможливорює відхилення від затвердженого базового профілю конфігурації (security baseline). Підхід, коли опис бажаної конфігурації зберігається у вигляді версійованих файлів і застосовується автоматизовано, є прямим попередником концепції Infrastructure as Code, докладніше розглянутої в розділі 14.

11.5. Резервне копіювання та відновлення служб

Жодна з розглянутих у цьому розділі служб не застрахована від відмови апаратного забезпечення, помилки адміністратора чи атаки шкідливого програмного забезпечення класу шифрувальників (ransomware). Тому невіддільною частиною адміністрування базових служб є регулярне **резервне копіювання** (бази служби каталогів, зони DNS, конфігурації DHCP) і, що не менш важливо, періодична **перевірка можливості відновлення** з резервної копії: копія, придатність якої до відновлення жодного разу не перевірялася, не є надійною гарантією.

Для критичних служб на практиці застосовують показники **RPO (Recovery Point Objective)**, максимально допустима втрата даних у часі, та **RTO (Recovery Time Objective)**, максимально допустимий час простою служби під час відновлення. Значення цих показників визначають вибір технології резервного копіювання (наприклад, необхідність реплікації контролера домену в реальному часі проти

нічного резервного копіювання) і безпосередньо пов'язані з питаннями відмовостійкості хмарної інфраструктури, розглянутими в розділі 14.

11.6. Автентифікація домену електронної пошти: SPF, DKIM, DMARC

Підрозділ 11.2 згадав SPF, DKIM і DMARC як доповнення до DNSSEC, що знижують ризик фішингу через підміну домену; розглянемо механізм дії кожного запису окремо, оскільки саме ці три записи, а не DNSSEC, є першим рубежем захисту поштового домену на практиці. **SPF (Sender Policy Framework)** — TXT-запис у зоні DNS домену, що перелічує IP-адреси серверів, яким дозволено надсилати пошту від імені цього домену; сервер отримувача, отримавши лист, перевіряє, чи входить адреса відправника до дозволеного переліку. **DKIM (DomainKeys Identified Mail)** додає до заголовка листа криптографічний підпис, обчислений закритим ключем відправника; сервер отримувача перевіряє цей підпис відкритим ключем, опублікованим у записі DNS домену, підтверджуючи, що вміст листа не було змінено після відправлення.

Окремо взяті SPF і DKIM лише надають отримувачу дані для перевірки, але не вказують, що робити, якщо перевірка не пройшла. Цю прогалину закриває **DMARC (Domain-based Message Authentication, Reporting and Conformance)**: політика, опублікована власником домену, яка явно визначає дію отримувача в разі невдалої перевірки SPF чи DKIM (none, спостереження без дій; quarantine, позначення як спам; reject, повна відмова від прийняття листа), а також механізм регулярних звітів, що дозволяє власнику домену бачити спроби розсилки від його імені сторонніми джерелами, зокрема під час фішингових кампаній, що видають себе за організацію.

11.7. Синхронізація часу: служба NTP та її роль у безпеці

Служба **NTP (Network Time Protocol)** синхронізує системний час усіх вузлів мережі з ієрархією еталонних джерела часу (рівні stratum, від атомних годинників і GPS-приймачів на найвищому рівні до внутрішніх серверів організації, що ретранслюють час нижчим рівням). На перший погляд, синхронізація годинника здається дрібницею адміністрування, що не стосується безпеки, однак вона є прихованою передумовою коректної роботи щонайменше двох механізмів, розглянутих раніше в цій книзі. По-перше, протокол Kerberos, на якому працює автентифікація в службі каталогів (підрозділ 11.1), відмовляє в автентифікації, якщо розбіжність годинників клієнта й контролера домену перевищує невеликий, суворо

заданий поріг (типово п'ять хвилин), саме як захист від атак повторного відтворення (replay attack). По-друге, кореляція подій із різних джерел у SIEM (розділ 3) і відтворення точної хронології інциденту при розслідуванні спираються на припущення, що мітки часу (timestamp) у журналах різних систем (розділ 2) взаємно узгоджені: розсинхронізований годинник хоча б одного сервера здатний спотворити встановлену послідовність подій атаки.

Оскільки коректність часу настільки критична, сам протокол NTP також потребує захисту: підміна джерела часу зломисником (атака на NTP) здатна або порушити автентифікацію Kerberos в усій мережі, або, витонченіше, змістити мітки часу в журналах саме настільки, щоб ускладнити розслідування. Розширення **NTS (Network Time Security)** додає до NTP автентифікацію джерела часу, аналогічно до того, як **DNSSEC** (підрозділ 11.2) додає автентифікацію до відповідей DNS.

11.8. Веб-проксі та фільтрація вихідного вебтрафіку

Поряд із базовими службами, розглянутими в підрозділах 11.1-11.3, більшість організацій адмініструє ще одну повсякденну, але часто недооцінену службу: **проксі-сервер вихідного вебтрафіку (forward proxy)**, через який проходять запити користувачів до зовнішніх вебресурсів. На відміну від міжмережевого екрана (підрозділ 10.4), що приймає рішення на рівні мережевих адрес і портів, веб-проксі працює на прикладному рівні й здатен приймати рішення на основі URL, категорії вебресурсу (наприклад, «азартні ігри», «соціальні мережі», «щойно зареєстровані домени», що часто корелюють із фішинговою чи шкідливою інфраструктурою) чи навіть вмісту сторінки.

Адміністрування проксі охоплює підтримання актуальних категорій фільтрації, керування винятками для бізнес-обґрунтованих потреб конкретних підрозділів, і, для HTTPS-трафіку, прийняття свідомого рішення щодо **інспекції зашифрованого трафіку (SSL/TLS inspection)**: проксі може розшифровувати HTTPS-з'єднання користувача (виступаючи довіреним посередником через власний внутрішній сертифікат, підрозділ 10.9), перевіряти вміст і повторно зашифровувати з'єднання до кінцевого ресурсу, що дає змогу застосувати ті самі методи детекції контенту, що й DLP (підрозділ 7.2), але водночас створює додаткові ризики приватності користувачів і продуктивності проксі-сервера, обговорені стосовно шифрованого трафіку загалом у підрозділі 6.5 модуля 1.

11.9. Файлові служби та служби друку: адміністрування спільних ресурсів

Найстаріші й найбуденніші з базових служб, спільні файлові ресурси (file shares) і служби друку, рідко потрапляють у фокус уваги при обговоренні кібербезпеки, хоча саме вони найчастіше стають першою точкою, звідки програма-шифрувальник (ransomware) поширює шкоду за межі одного скомпрометованого вузла: зашифрувавши локальні файли, шкідливе програмне забезпечення зазвичай продовжує сканувати й шифрувати всі мережеві диски, до яких скомпрометований обліковий запис має доступ на запис.

Адміністрування файлових служб охоплює розмежування прав доступу на рівні спільного ресурсу й на рівні файлової системи одночасно (обидва рівні діють сумісно, і фактичний доступ визначається найсуворішим з них), регулярний перегляд списків доступу (аналогічно до аудиту правил міжмережевого екрана з підрозділу 10.8, але для файлової системи), і, дедалі частіше, впровадження **файлів-приманок (canary files)**, фальшивих файлів у спільних ресурсах, чия несподівана зміна чи шифрування служить раннім індикатором активності шифрувальника, за принципом, спорідненим із honeypot (підрозділ 6.7 модуля 1). Служба друку, своєю чергою, менш очевидна мішень, але історично неодноразово ставала джерелом вразливостей самого протоколу друку, що дозволяли віддалене виконання коду на сервері друку з правами системи, тому вимагає такого самого дисциплінованого патч-менеджменту (підрозділ 9.8), як і будь-яка інша служба цього розділу.

11.10. Основи захисту серверів баз даних

Базові служби, розглянуті в підрозділах 11.1-11.9, готують інфраструктурне середовище, у якому зрештою працює найцінніший, з погляду зловмисника, ресурс організації, сервер бази даних. Базовий захист СУБД спирається на ті самі принципи, що вже розглядалися стосовно інших служб цього розділу, застосовані до специфіки бази даних: розмежування прав доступу на рівні окремої таблиці чи навіть стовпця, а не лише на рівні всієї бази (деталізація принципу найменших привілеїв, підрозділ 9.4); окремий, непривілейований обліковий запис для кожного застосунку, що звертається до бази, замість спільного облікового запису адміністратора; і шифрування даних як у стані спокою (encryption at rest), так і при передаванні (у поєднанні з TLS, підрозділ 13.5).

Специфічним для баз даних ризиком, не характерним для файлових служб чи служби друку, є **SQL-ін'єкція**: клас атак, коли зловмисник вбудовує шкідливий SQL-код у поле вводу вебзастосунку, змушуючи базу даних виконати

незапланований запит. Захист від цього класу атак лежить переважно на рівні розробки застосунку (параметризовані запити), а не адміністрування СУБД як такого, однак адміністратор бази даних відповідає за суміжний, доповнювальний захід: журналювання аномальних запитів (підрозділ 2.9 модуля 1) і регулярний аудит цілісності критичних таблиць (розділ 21 модуля 3) як останній рубіж виявлення вже здійсненої, але ще не поміченої атаки.

Лабораторна робота 11. Розгортання контрольованого середовища DNS/DHCP та аналіз атаки rogue DHCP

Мета роботи: отримати практичні навички адміністрування служб DNS і DHCP та розпізнавання ознак несанкціонованого DHCP-сервера в мережі.

Хід роботи:

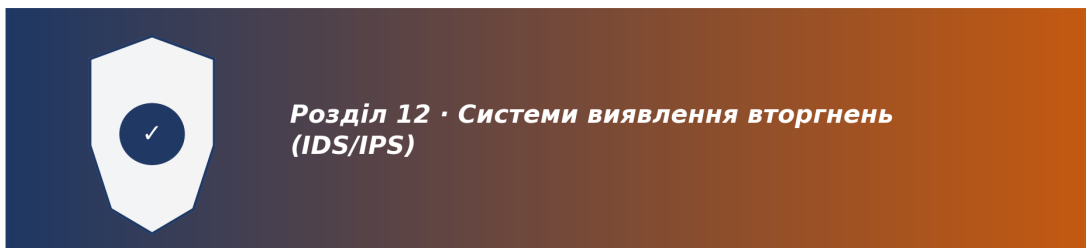
1. розгорнути в ізолюваному тестовому середовищі (віртуальні машини) контролер домену з інтегрованою службою DNS та окремий DHCP-сервер;
2. налаштувати діапазон видачі адрес, резервування для одного тестового вузла та зареєструвати кілька DNS-записів вручну і через автоматичну реєстрацію;
3. розгорнути в тому самому сегменті другий, «неавторизований» DHCP-сервер із свідомо неправильним шлюзом за замовчуванням та зафіксувати засобами аналізу трафіку (з розділу 6) появу конкуруючих DHCP-відповідей;
4. запропонувати й, за наявності керованого комутатора або його емуляції, застосувати захід протидії (DHCP snooping або еквівалент);
5. оформити звіт із порівнянням поведінки мережі до та після застосування захисного заходу.

Контрольні питання до розділу 11

1. Чому компрометація контролера домену вважається однією з найкритичніших подій для всієї ІТС?
2. Яку задачу вирішує DNSSEC і від якого класу атак він захищає?
3. Опишіть механізм атаки DHCP starvation і атаки rogue DHCP server. Чим вони відрізняються?
4. Яким технічним заходом на рівні комутатора протидіють несанкціонованому DHCP-серверу?
5. Що означають показники RPO і RTO і як вони впливають на вибір стратегії резервного копіювання?

6. Яку роль виконує контролер домену у взаємодії базових інформаційних служб (рисунок 11.1)?
7. Що таке tier-модель адміністрування контролерів домену і навіщо вона потрібна?
8. Чим DNS, інтегрована зі службою каталогів, відрізняється від автономного DNS-сервера?
9. Які записи (SPF, DKIM, DMARC) доповнюють DNSSEC для зниження ризику фішингових атак через підміну домену?
10. Опишіть механізм видачі адреси службою DHCP та основні параметри, що передаються клієнту.
11. Чим атака rogue DHCP server відрізняється від атаки DHCP starvation за метою зломисника?
12. Що таке групові політики (Group Policy) і яку задачу централізованого управління вони вирішують?
13. Наведіть приклад аналога Group Policy в іншій екосистемі (Linux/хмара).
14. Що таке базовий профіль конфігурації (security baseline) і як централізоване управління конфігурацією його забезпечує?
15. Наведіть приклад, як цільові значення RPO і RTO впливають на вибір технології резервного копіювання.
16. Чому резервну копію необхідно періодично перевіряти на придатність до відновлення, а не лише створювати за розкладом?

РОЗДІЛ 12. СИСТЕМИ ВИЯВЛЕННЯ ВТОРГНЕНЬ (IDS/IPS): АДМІНІСТРУВАННЯ



Зміст розділу

12.1. Класифікація систем виявлення вторгнень

12.2. Розміщення IDS та IPS у мережі

12.3. Методи виявлення: сигнатурні, аномалій, гібридні

12.4. Огляд поширених рішень з відкритим кодом

12.5. Адміністрування правил і зниження частки хибних спрацювань

12.6. Методи ухилення від систем виявлення вторгнень

12.7. Продуктивність і відмовостійкість IPS-рішень

12.8. EDR/XDR: виявлення на кінцевій точці як доповнення до мережевого IDS/IPS

12.9. Перевірка ефективності виявлення: адверсарна симуляція та purple team

12.10. NDR (Network Detection and Response) як доповнення до IDS/IPS

12.1. Класифікація систем виявлення вторгнень

Системи виявлення й запобігання вторгненням класифікують передусім за розташуванням точки контролю. **NIDS (Network Intrusion Detection System)** аналізує копію мережевого трафіку, спостерігаючи за подіями на рівні сегмента мережі. **HIDS (Host Intrusion Detection System)** встановлюється на конкретному вузлі (сервері, робочій станції) і аналізує системні виклики, журнали, цілісність файлів на цьому вузлі. Ці два підходи доповнюють, а не замінюють один одного: NIDS не бачить дій, що відбуваються локально на вузлі й не проходять мережею, тоді як HIDS не має видимості мережевого трафіку інших вузлів.

Другою віссю класифікації є режим дії. **IDS (Intrusion Detection System)** працює пасивно: виявляє й сигналізує про підозрілу активність, але не втручається в трафік. **IPS (Intrusion Prevention System)** працює активно, безпосередньо в розриві каналу передавання даних, і здатна заблокувати шкідливий пакет чи з'єднання в реальному часі, до того як воно досягне мети.

12.2. Розміщення IDS та IPS у мережі

Відмінність між пасивним і активним режимом роботи безпосередньо визначає спосіб фізичного чи логічного підключення системи до мережі. Рисунок 12.1 зображує обидва варіанти одночасно для порівняння.

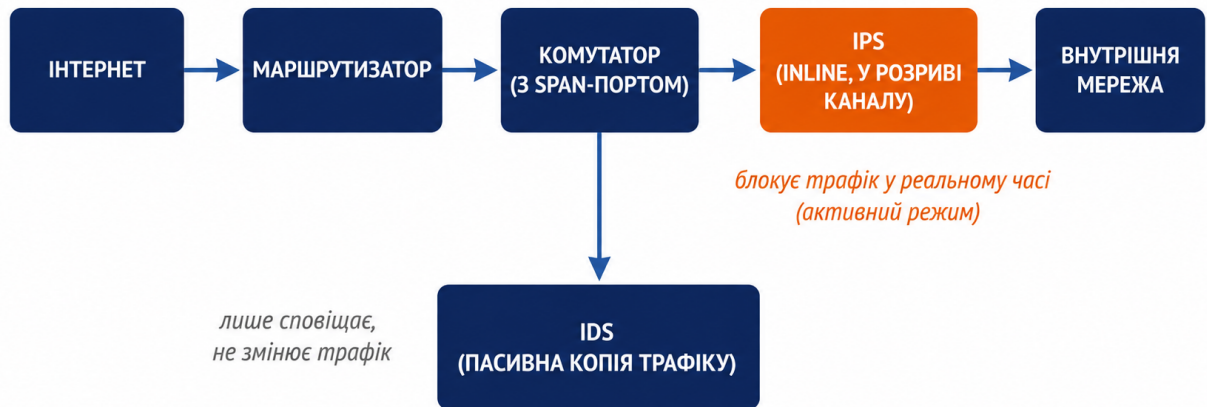


Рис. 12.1. Розміщення IDS (пасивний режим) та IPS (активний, inline-режим) у мережі

Джерело: розроблено автором

IDS зазвичай підключають до дзеркального (SPAN, mirror) порту комутатора або через мережевий TAP: система отримує копію трафіку, аналізує її й формує сповіщення, але фізично не здатна вплинути на оригінальний потік даних навіть у разі виявлення атаки, а помилка чи відмова самої IDS не впливає на роботу мережі. **IPS**, навпаки, встановлюється безпосередньо в розриві каналу, тобто увесь трафік фізично проходить крізь пристрій IPS. Це дає змогу заблокувати шкідливий пакет до моменту досягнення ним мети, але водночас означає, що відмова або перевантаження самого IPS може порушити роботу легітимного трафіку, а тому продуктивність та відмовостійкість пристрою IPS є значно критичнішими вимогами, ніж для IDS.

12.3. Методи виявлення: сигнатурні, аномалій, гібридні

Сигнатурний метод порівнює мережевий трафік або поведінку системи з базою відомих шаблонів атак (сигнатур). Метод дає високу точність і майже не створює хибнопозитивних спрацювань для відомих загроз, але за визначенням безсилий проти атак, сигнатури яких ще не додано до бази, включно з атаками нульового дня.

Метод виявлення аномалій будує статистичну модель нормальної поведінки мережі чи системи (базову лінію) і сигналізує про суттєве відхилення від неї, не потребуючи заздалегідь відомої сигнатури конкретної атаки. Ціною цієї гнучкості є вища частка хибнопозитивних спрацювань: легітимна, але нетипова активність (наприклад, разове масове архівування файлів перед плановим звітним періодом) також може бути позначена як аномалія.

На практиці більшість сучасних комерційних та відкритих рішень поєднують обидва підходи в **гібридній моделі**, доповнюючи сигнатурну базу модулями виявлення аномалій, а дедалі частіше й методами машинного навчання; питання застосування штучного інтелекту до аналізу подій безпеки докладно розглянуто в модулі 3.

12.4. Огляд поширених рішень з відкритим кодом

Серед систем виявлення вторгнень з відкритим вихідним кодом історично найбільшого поширення набули **Snort** і його архітектурний наступник **Suricata**, обидва працюють переважно за сигнатурним принципом і використовують сумісний формат правил, а також **Zeek** (раніше відомий як Bro), що зорієнтований не стільки на генерацію сигнатурних сповіщень, скільки на глибокий структурований журнал усіх мережевих сесій, придатний для подальшого аналізу в SIEM-системі (розділ 3). **Suricata**, на відміну від однопотокового за задумом **Snort**, спроєктована з підтримкою багатопотокової обробки, що дає їй перевагу в продуктивності на каналах з високою пропускною здатністю, а також підтримує апаратне прискорення обробки пакетів.

Вибір між цими рішеннями на практиці визначається наявною інфраструктурою: **Suricata** добре інтегрується як джерело подій у **Sentinel** і **Splunk** (розділи 4-5) завдяки структурованому виводу подій у форматі EVE JSON, тоді як **Zeek** частіше застосовують як додаткове, а не основне джерело виявлення, саме завдяки глибині журналювання мережевої взаємодії.

12.5. Адміністрування правил і зниження частки хибних спрацювань

Розгортання IDS/IPS «з коробки», без подальшого адміністрування, типово призводить до одного з двох небажаних сценаріїв: або система генерує стільки сповіщень, що аналітик фізично не встигає їх опрацьовувати (alert fatigue, вже згадана в розділі 1), або, у режимі IPS, надто агресивний набір правил починає блокувати легітимний трафік. Основними інструментами адміністратора для запобігання цим сценаріям є: **тюнінг порогових значень** правил під конкретне

середовище; **придушення (suppression)** сповіщень для джерел, свідомо відомих як легітимні (наприклад, власний сканер вразливостей з розділу 1); **регулярне оновлення бази сигнатур; написання власних правил** під специфічні для організації загрози, яких немає в стандартній базі постачальника.

Типове правило Suricata/Snort складається з дії (alert, drop), протоколу, адрес і портів джерела та призначення, а також опцій, що визначають конкретну ознаку, яку слід шукати в пакеті, наприклад сигнатуру відомого шкідливого корисного навантаження чи характерний рядок у заголовку HTTP-запиту. Розробка й тестування власного правила є предметом практикуму до цього розділу.

12.6. Методи ухилення від систем виявлення вторгнень

Зловмисник, що знає про наявність IDS/IPS у мережі жертви, зазвичай намагається не просто атакувати, а атакувати непомітно для системи виявлення; сукупність таких прийомів називають **ухиленням (evasion)**. До поширених технік належать: **фрагментація пакетів**, навмисний поділ шкідливого корисного навантаження на дрібні фрагменти таким чином, щоб жоден окремих фрагмент не відповідав сигнатурі, а збирання цілісного повідомлення відбувалося лише на боці кінцевого вузла; **маніпуляція часовими інтервалами (timing attacks)**, розтягування дій атаки на тривалий час, щоб вони не перевищили пороги виявлення аномалій (підрозділ 12.3); **кодування й обфускація корисного навантаження** (наприклад, нестандартне кодування символів URL) у спосіб, що зберігає працездатність для кінцевого застосунку, але не збігається з сигнатурою дослівно; і, найбільш системно, **використання шифрованих каналів (TLS)**, розглянуте детальніше в підрозділі 6.5 модуля 1, яке взагалі позбавляє сигнатурний аналіз доступу до вмісту трафіку.

Практичний висновок для адміністратора IDS/IPS полягає в тому, що жодна сигнатурна база не забезпечує повного захисту від цілеспрямованого й обізнаного супротивника: гібридна модель, що поєднує сигнатури з аналізом аномалій (підрозділ 12.3), а також своєчасне оновлення й нормалізація (пересклеювання фрагментованих пакетів перед аналізом, TCP/IP reassembly) значно ускладнюють, хоча й не унеможливають повністю, застосування техніки ухилення.

12.7. Продуктивність і відмовостійкість IPS-рішень

Підрозділ 12.2 уже зазначив, що відмова IPS, на відміну від відмови IDS, здатна порушити роботу легітимного трафіку, оскільки весь трафік фізично

проходить крізь пристрій. Це висуває дві додаткові вимоги до адміністрування IPS, не актуальні для пасивної IDS. По-перше, **пропускна здатність** пристрою IPS повинна мати запас відносно пікового навантаження каналу, оскільки глибока інспекція пакетів (підрозділ 6.1 модуля 1) є обчислювально вартісною операцією, а перевантажений IPS стає вузьким місцем мережі. По-друге, необхідно свідомо визначити поведінку пристрою в разі його власного збою: режим **fail-open** пропускає трафік без інспекції у разі відмови IPS, зберігаючи доступність мережі коштом тимчасової втрати захисту, тоді як режим **fail-closed** повністю блокує трафік у разі відмови, зберігаючи захист коштом повної недоступності мережі. Вибір між цими режимами є рішенням не суто технічним, а управлінським, що узгоджується з вимогами RTO/RPO (підрозділ 11.5) і критичністю сегмента мережі, який захищає конкретний пристрій.

12.8. EDR/XDR: виявлення на кінцевій точці як доповнення до мережевого IDS/IPS

Підрозділ 12.1 розрізняє NIDS і HIDS за точкою розташування; сучасний розвиток концепції HIDS для кінцевих точок відомий під назвою **EDR (Endpoint Detection and Response)**. На відміну від класичного антивірусу, орієнтованого переважно на сигнатурне виявлення вже відомого шкідливого файлу, EDR безперервно записує детальну телеметрію процесів, мережних з'єднань і змін у файловій системі кінцевої точки (концептуально спираючись на ту саму поглиблену видимість, яку дає Sysmon чи auditd, розглянуті в підрозділі 2.9 модуля 1), і застосовує до цього потоку поведінковий аналіз, здатний виявити підозрілу послідовність дій навіть тоді, коли жоден окремий файл не збігається з відомою сигнатурою. Що не менш важливо, EDR дає адміністратору й засіб реагування безпосередньо з консолі: ізолювати кінцеву точку від мережі, примусово завершити підозрілий процес, відкотити зміни, внесені шкідливим програмним забезпеченням.

XDR (Extended Detection and Response), вже згаданий побіжно в підрозділі 4.1 модуля 1 стосовно екосистеми Microsoft Defender, розширює логіку EDR за межі окремої кінцевої точки, об'єднуючи телеметрію з кінцевих точок, мережі (тобто самих NIDS/IPS цього розділу) і хмарних сервісів в єдиний контур кореляції, за задумом постачальника, тіснішої й глибшої, ніж загальна кореляція SIEM (розділ 3 модуля 1), але, на відміну від SIEM, здебільшого обмеженої продуктами одного постачальника. Практичний висновок для адміністратора мережевого IDS/IPS: мережеве виявлення й EDR/XDR не конкурують, а доповнюють одне одного, покриваючи ту саму слідчу картину атаки з двох різних, взаємодоповнюючих точок спостереження, мережевого трафіку та поведінки самого кінцевого вузла.

12.9. Перевірка ефективності виявлення: адверсарна симуляція та purple team

Наявність увімкненого й формально налаштованого IDS/IPS (підрозділи 12.4-12.5) не гарантує, що система дійсно виявить реальну атаку: єдиний спосіб це перевірити, це змодельовати атаку в контрольованих умовах і подивитися, чи спрацює виявлення. **Red team** (команда, що імітує дії реального зловмисника, часто без попередження команди захисту) і **purple team** (формат, у якому команда атаки й команда захисту працюють спільно, у реальному часі обговорюючи, яка дія атаки була чи не була виявлена і чому) є двома усталеними форматами такої перевірки. На відміну від тестування на проникнення (penetration testing), зосередженого передусім на пошуку самих вразливостей, адверсарна симуляція перевіряє саме здатність команди моніторингу й засобів виявлення, розглянутих у цьому розділі, вчасно помітити й правильно відреагувати на реалістичний ланцюг атаки.

Практичним інструментом планування такої перевірки знову слугує матриця MITRE ATT&CK (підрозділ 8.4 модуля 1): команда заздалегідь узгоджує, які конкретні техніки будуть відтворені (наприклад, конкретний метод латерального руху з підрозділу 6.8 модуля 1), а після завершення вправи зіставляє відтворені техніки з тим, що реально зафіксували IDS/IPS і SIEM, формуючи список прогалин виявлення (detection gaps), які й стають вхідними даними для наступного циклу тюнінгу правил (підрозділ 3.8 модуля 1) чи написання нових власних сигнатур (підрозділ 12.5).

12.10. NDR (Network Detection and Response) як доповнення до IDS/IPS

Сигнатурні та навіть аномалійні IDS/IPS (підрозділи 12.2-12.5), розглянуті вище, працюють переважно на рівні окремого пакета чи сесії й погано бачать повільну, розподілену в часі активність зловмисника всередині вже скомпрометованої мережі. **NDR (Network Detection and Response)** доповнює цю картину, безперервно аналізуючи метадані мережевого трафіку (NetFlow, IPFIX, повні пакетні захоплення на ключових точках) за допомогою поведінкових моделей і машинного навчання, щоб виявляти аномалії на кшталт незвичного бічного руху (lateral movement), повільного витoku даних невеликими порціями чи спілкування з командно-контрольною інфраструктурою через легітимні на вигляд протоколи. На відміну від EDR (підрозділ 12.8), що спостерігає за кінцевою точкою зсередини, NDR дає мережеву видимість там, де EDR-агент відсутній: у сегментах з IoT/OT-пристроями, застарілими системами чи некерованим обладнанням, що є типовою прогалиною в гібридних мережах, розглянутих у розділі 14.

Практична цінність NDR розкривається саме в поєднанні з іншими джерелами моніторингу цього модуля: підозрілий мережевий патерн, виявлений NDR, стає приводом підняти відповідні журнали з SIEM (розділ 3 модуля 1) і перевірити стан конкретного хоста через EDR, замикаючи цикл кореляції за принципом XDR (підрозділ 12.8). Такий підхід узгоджується з моделлю «припущення про компрометацію» (assume breach), яка лежить в основі Zero Trust архітектури (підрозділ 10.9): оскільки периметровий захист не гарантує відсутності проникнення, безперервний поведінковий моніторинг внутрішнього трафіку через NDR стає обов'язковим доповненням до традиційного IDS/IPS, а не опціональним інструментом.

Практикум 12. Написання та тестування власного правила Suricata

Мета роботи: набути навичок адміністрування системи виявлення вторгнень на прикладі написання, розгортання й перевірки власного сигнатурного правила.

Хід роботи:

1. розгорнути Suricata в тестовому середовищі в пасивному режимі (IDS) на інтерфейсі, що отримує копію трафіку тестового сегмента;
2. ознайомитися зі структурою стандартного правила зі складу відкритої бази (Emerging Threats Open) та визначити призначення кожного поля;
3. написати власне правило, що виявляє характерну ознаку узгодженої з викладачем умовної підозрілої активності (наприклад, звернення до певного шляху URL або передавання певного рядка в корисному навантаженні);
4. відтворити відповідний трафік у тестовому середовищі та підтвердити спрацювання правила за журналом сповіщень;
5. свідомо викликати хибнопозитивне спрацювання, підібрати умову, що відрізняє легітимний трафік від цільового, і скоригувати правило для його усунення.

Примітка для викладача. Актуальний синтаксис правил і перелік підтримуваних ключових слів варто звіряти з чинною документацією проєкту Suricata на момент проведення заняття, оскільки мова правил поступово розширюється.

Контрольні питання до розділу 12

1. Чим відрізняються NIDS і HIDS за точкою контролю та які загрози кожен з них здатен пропустити?
2. Поясніть різницю в мережевому підключенні між IDS (пасивний режим) та IPS (inline-режим) і наслідки цієї різниці для відмовостійкості мережі.
3. Порівняйте сигнатурний метод і метод виявлення аномалій за точністю та здатністю виявляти нові атаки.
4. Чим Suricata архітектурно відрізняється від Snort?
5. Якими засобами адміністратор може знизити частку хибнопозитивних спрацювань системи виявлення вторгнень, не втрачаючи здатності виявляти реальні атаки?
6. Яку задачу вирішує NIDS, яку не може вирішити HIDS, і навпаки?
7. Опишіть, чому відмова IPS може вплинути на роботу легітимного трафіку, а відмова IDS - ні (рисунок 12.1).

8. Що таке SPAN-порт (або мережевий TAP) і яку роль він відіграє для розгортання IDS?
9. У чому перевага сигнатурного методу виявлення атак і в чому його головне обмеження?
10. Чим атака нульового дня становить проблему саме для сигнатурного підходу?
11. Наведіть приклад того, як метод виявлення аномалій може дати хибнопозитивне спрацювання на легітимну активність.
12. Яку унікальну перевагу дає Zeek порівняно з класичними сигнатурними системами?
13. Що таке формат EVE JSON і чому він зручний для інтеграції Suricata з SIEM?
14. Перелічіть щонайменше три інструменти адміністратора для зниження частки хибнопозитивних спрацювань IDS/IPS.
15. З яких основних полів складається типове правило Snort/Suricata?
16. Наведіть приклад придушення (suppression) сповіщення для джерела, свідомо відомого як легітимне.

РОЗДІЛ 13. ПОБУДОВА ЗАХИЩЕНИХ МЕРЕЖ ВІДПОВІДНО ДО МОДЕЛІ OSI



Зміст розділу

- 13.1. Модель OSI як основа аналізу захисту мережі**
- 13.2. Захист на фізичному, канальному та мережевому рівнях**
- 13.3. Програмно-конфігуровані мережі (SDN) як сучасний підхід до побудови захищених мереж**
- 13.4. Захист на транспортному, сеансовому та прикладному рівнях**
- 13.5. TLS у деталях: рукописання та перехід від версії 1.2 до 1.3**
- 13.6. Мікросегментація мережі на основі SDN**
- 13.7. Захист бездротових мереж (Wi-Fi)**

13.8. VLAN і трункінг: технічна реалізація сегментації канального рівня

13.9. Мережева документація: схеми та реєстр IP-адрес (IPAM)

13.1. Модель OSI як основа аналізу захисту мережі

Семирівнева еталонна модель взаємодії відкритих систем (OSI, Open Systems Interconnection) традиційно використовується для опису передавання даних мережею, але вона однаково корисна як структурована рамка для аналізу захищеності: кожен рівень має власні протоколи, а отже, і власні характерні вразливості та засоби захисту, які недоцільно змішувати між собою. Таблиця 13.1 і рисунок 13.1 узагальнюють типові заходи захисту, застосовні на кожному рівні моделі.

Таблиця 13.1. Типові загрози та засоби захисту на рівнях моделі OSI

Рівень OSI	Типова загроза	Основний засіб захисту
7. Прикладний	SQL-ін'єкція, XSS, логічні вразливості застосунку	WAF, безпечна розробка (SSDLC), аутентифікація
6. Представлення	підміна формату даних, недоліки серіалізації	TLS/SSL, шифрування форматів даних
5. Сеансовий	перехоплення чи фіксація сесії (session hijacking)	керування сесіями, таймаути, безпечні токени
4. Транспортний	перехоплення сесії, сканування портів	TLS, обмеження й фільтрація портів, firewall рівня L4
3. Мережевий	підміна IP-адреси, атаки маршрутизації	IPsec, списки контролю доступу (ACL), IDS/IPS
2. Канальний	ARP spoofing, MAC-флудинг, VLAN hopping	802.1X, port security, статична прив'язка ARP
1. Фізичний	несанкціонований доступ до кабелю чи порту	контроль доступу до приміщень, відключення невикористаних портів

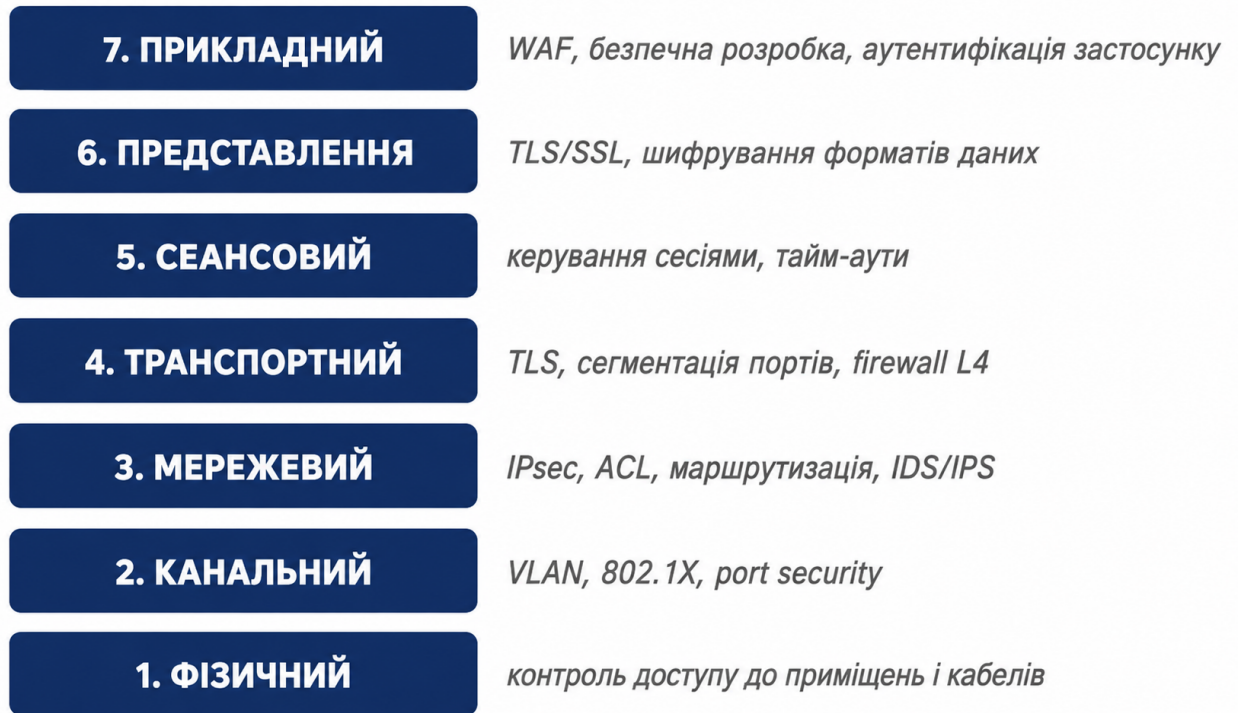


Рис. 13.1. Заходи захисту, розподілені за рівнями моделі OSI

Джерело: розроблено автором

Практична цінність такого пошарового аналізу для адміністратора безпеки полягає в тому, що він унеможливорює типову помилку, коли всі зусилля зосереджуються на одному рівні (найчастіше мережевому чи прикладному), залишаючи каналний чи фізичний рівень без уваги, хоча атака на кожен із них здатна повністю компрометувати захищеність усієї системи незалежно від якості захисту вищих рівнів.

13.2. Захист на фізичному, каналному та мережевому рівнях

На **фізичному рівні** захист зводиться до контролю фізичного доступу: замкнені серверні приміщення, відключення невикористаних портів комутатора, застосування шафи з контролем доступу для мережевого обладнання. На **каналному рівні** основну загрозу становить довіра, яку протоколи цього рівня апріорі виявляють до пристроїв усередині сегмента: протокол ARP не передбачає автентифікації, що уможливорює атаку ARP spoofing, підміну відповідності MAC- та IP-адреси з метою перехоплення трафіку. Протидією є технологія **802.1X** (автентифікація пристрою до отримання мережевого доступу), функція **port security** (обмеження кількості або переліку MAC-адрес, дозволених на порту комутатора) та

захист від VLAN hopping шляхом явного, а не автоматичного узгодження транкових портів.

На **мережевому рівні** основними засобами захисту є списки контролю доступу (ACL) на маршрутизаторах, протокол **IPsec** для захисту трафіку між мережевими сегментами (зокрема, для VPN-тунелів типу site-to-site), а також розглянуті в розділі 12 системи виявлення й запобігання вторгненням. Специфічною для мережевого рівня загрозою є підміна маршрутизації: без додаткового захисту протоколи динамічної маршрутизації (наприклад, OSPF, BGP) можуть прийняти неправдиву інформацію про маршрути від зловмисника, що видає себе за легітимний маршрутизатор, тому автентифікація сусідів маршрутизації є обов'язковою вимогою для мереж, чутливих до безпеки.

13.3. Програмно-конфігуровані мережі (SDN) як сучасний підхід до побудови захищених мереж

Традиційна мережева архітектура, розглянута у підрозділах 13.1-13.2, передбачає, що площа керування (яка ухвалює рішення про маршрутизацію й фільтрацію) та площа передавання даних (яка фізично пересилає пакети) суміщені в кожному окремому пристрої. Концепція **програмно-конфігурованих мереж (Software-Defined Networking, SDN)** фізично розділяє ці дві площини: рішення про керування трафіком централізовано ухвалює контролер SDN, тоді як комутатори й маршрутизатори лише виконують його інструкції, найпоширеніше за протоколом **OpenFlow**.

Для захищеності мережі централізація керування в SDN є одночасно і перевагою, і новим класом ризику. Перевагою є те, що політики безпеки (сегментація, списки доступу, реагування на виявлену загрозу) можна змінювати одночасно для всієї мережі з єдиної точки керування, а не послідовно на кожному пристрої окремо, і саме тому SDN розглядають як перспективну основу для централізованого моніторингу мережі та автоматизованого реагування, включно з децентралізованим розгортанням функцій моніторингу безпосередньо на мережевих пристроях і швидким (порядку десятків мілісекунд) автоматичним відновленням зв'язності мережі після відмови вузла без участі контролера, як показано в [9] зі списку джерел цього розділу. Ризиком, натомість, є те, що сам контролер SDN стає точкою, компрометація якої означає контроль над усією мережею, а тому захист каналу між контролером і комутаторами та автентифікація самого контролера є обов'язковою умовою безпечного впровадження SDN, а не опціональним

доповненням. Окремою прикладною задачею є саме моніторинг коректності керуючих повідомлень протоколу OpenFlow між контролером і комутаторами: підходи до вимірювання мережі, що дозволяють виявляти неправильну чи зловмисно підмінену взаємодію на цьому рівні ще до її наслідків для площини передавання даних, розглянуто в [7] зі списку джерел цього розділу.

13.4. Захист на транспортному, сеансовому та прикладному рівнях

На **транспортному рівні** основним засобом захисту лишається шифрування каналу за допомогою **TLS**, що прийшов на зміну застарілому й вразливому **SSL**; сучасні рекомендації (зокрема NIST SP 800-52) вимагають використання **TLS** версії 1.2 як мінімально прийнятної та рекомендують перехід на **TLS 1.3**. Додатково на цьому рівні застосовують обмеження діапазону дозволених портів засобами міжмережевого екрана, розглянутого в розділі 10. На **сеансовому рівні** захист зосереджується на керуванні життєвим циклом сесії користувача: обмежену часі бездіяльності (тайм-ауті), безпечній генерації та зберіганні токенів сесії, примусовому завершенні сесії після зміни пароля чи виявлення підозрілої активності.

Прикладний рівень акумулює найбільшу кількість специфічних вразливостей, оскільки саме на ньому працює безпосередньо бізнес-логіка застосунку: ін'єкції (**SQL**, командні), міжсайтовий скриптинг (**XSS**), недоліки авторизації на рівні окремих об'єктів даних. Основними заходами захисту є **міжмережевий екран рівня застосунків (Web Application Firewall, WAF)**, що фільтрує **HTTP(S)**-запити за поведінковими та сигнатурними ознаками атак, і, що важливіше в довгостроковій перспективі, впровадження практик безпечного життєвого циклу розробки (**Secure SDLC**): аналіз коду на вразливості ще на етапі розробки, а не виключно захист уже розгорнутого застосунку ззовні.

13.5. TLS у деталях: рукописання та перехід від версії 1.2 до 1.3

Підрозділ 13.4 назвав **TLS** основним засобом захисту транспортного рівня; розглянемо механізм його роботи детальніше, оскільки саме структура початкового обміну повідомленнями (**ClientHello**, **ServerHello**), яка й дає змогу побудувати **TLS**-фінгерпринт **JA3**, згаданий у підрозділі 6.5 модуля 1, лежить в основі протоколу. Встановлення з'єднання складається з кількох кроків: клієнт і сервер узгоджують версію протоколу й набір криптографічних алгоритмів (cipher suite); сервер надає сертифікат, за яким клієнт перевіряє автентичність сервера (та сама інфраструктура

сертифікатів, що лежить в основі DNSSEC і DKIM, розглянутих у підрозділах 11.2 і 11.6); сторони узгоджують спільний сеансовий ключ методом асиметричного обміну ключами, після чого решта сеансу шифрується вже значно швидшим симетричним алгоритмом.

Перехід від TLS 1.2 до рекомендованої нині версії **1.3** (питання, поставлене в контрольному питанні 13 до цього розділу) не є суто косметичним оновленням номера версії. По-перше, TLS 1.3 скорочує кількість раундів обміну повідомленнями, необхідних для встановлення з'єднання (з двох раундів у 1.2 до одного, а за повторного з'єднання з відомим сервером, можливо, і без додаткового раунду), що напряду зменшує затримку. По-друге, специфікація TLS 1.3 повністю виключає застарілі й криптографічно ослаблені алгоритми, дозволені (хоча й не обов'язкові) у TLS 1.2, зокрема статичний обмін ключами RSA без властивості прямої секретності (forward secrecy), за якої компрометація довгострокового закритого ключа сервера в майбутньому не дає змоги розшифрувати раніше перехоплений трафік. Саме тому NIST SP 800-52, згаданий у підрозділі 13.4, наполягає на переході до TLS 1.3, а не лише допускає TLS 1.2 як тимчасово прийнятний мінімум.

13.6. Мікросегментація мережі на основі SDN

Підрозділ 10.4 представив сегментацію мережі на рівні VLAN і підмереж: групування вузлів за призначенням у порівняно великі логічні блоки. **Мікросегментація** доводить цей самий принцип до граничної деталізації, окремої політики доступу для кожного окремого робочого навантаження (workload) чи навіть окремого застосунку, незалежно від того, в якому фізичному сегменті мережі він фактично розташований. Традиційна мережева архітектура робить таку деталізацію практично нездійсненною: кожна дрібна зміна політики вимагала б перенастроювання окремих комутаторів.

Централізована площина керування SDN, розглянута в підрозділі 13.3, знімає це обмеження: політика мікросегментації описується один раз на контролері й застосовується узгоджено для всієї мережі, включно з динамічним оновленням у разі переміщення робочого навантаження між фізичними вузлами, наприклад під час міграції віртуальної машини. Мікросегментація на основі SDN є, по суті, мережевою реалізацією принципу Zero Trust Network Access, розглянутого в підрозділі 10.6: замість довіри «за замовчуванням» усередині великого сегмента кожен потік трафіку між окремими навантаженнями має бути явно дозволений політикою. Той самий

принцип, застосований не до фізичної чи віртуалізованої мережі, а до контейнерів, розглядається в підрозділі 15.5 стосовно мережевих політик Kubernetes.

13.7. Захист бездротових мереж (Wi-Fi)

Заходи захисту каналного рівня, розглянуті в підрозділі 13.2 для дротової мережі (802.1X, port security), мають власний, специфічний відповідник для бездротового середовища, де сам носій передавання, радіоефір, у принципі доступний будь-кому в зоні покриття, а не лише пристроям, фізично підключеним до кабелю. Еволюція протоколів захисту Wi-Fi пройшла шлях від застарілого й криптографічно зламаного WEP через WPA і WPA2 (обидва засновані на протоколі TKIP чи AES-CCMP) до сучасного **WPA3**, що усуває низку відомих недоліків попередників, зокрема вразливість до офлайн-підбору пароля через захоплений початковий обмін (характерну для WPA2-Personal), замінюючи механізм узгодження спільного ключа на протокол SAE (Simultaneous Authentication of Equals), стійкіший до такої атаки.

Практичне адміністрування безпеки Wi-Fi виходить за межі самого вибору протоколу шифрування: розмежування мереж за призначенням через окремі **SSID** (наприклад, окрема мережа для співробітників з доступом до внутрішніх ресурсів, окрема гостьова мережа з доступом лише в інтернет, кожна прив'язана до власного VLAN, підрозділ 13.8) реалізує для бездротового середовища той самий принцип сегментації, що вводився в підрозділі 10.4; а для корпоративного середовища замість єдиного спільного пароля мережі (WPA-Personal) застосовують **WPA-Enterprise** з індивідуальною автентифікацією кожного користувача через RADIUS (таблиця 10.1 з підрозділу 10.3), що дає змогу відкликати доступ конкретного співробітника без зміни пароля для всіх інших.

Практична цінність своєчасної міграції з WEP/WPA на WPA3-Enterprise не обмежується суто технічною стійкістю до злому: дослідження Rudin, Vasylenko, Grynkevych, присвячене саме еволюції захисту Wi-Fi у фінансових установах, показує, що вартість своєчасної міграції протоколу систематично, на порядки, нижча за економічні наслідки одного успішного інциденту, спричиненого застарілим протоколом шифрування, що є додатковим, суто управлінським аргументом на користь пріоритизації цього оновлення в плані патч-менеджменту (підрозділ 9.8).

13.8. VLAN і трункінг: технічна реалізація сегментації каналного рівня

Підрозділи 10.4 і 13.7 неодноразово посилалися на VLAN як засіб сегментації, не розкриваючи механізму його роботи; розглянемо технічну реалізацію. **VLAN (Virtual Local Area Network)** дозволяє логічно розділити один фізичний комутатор (і навіть кілька комутаторів, з'єднаних між собою) на кілька незалежних широкомовних доменів, кожен з яких поводить себе як окрема фізична мережа, хоча трафік різних VLAN може проходити через ту саму фізичну інфраструктуру. Порт комутатора, до якого підключений кінцевий пристрій, налаштовують у режимі **access port**, прив'язаному до одного конкретного VLAN, тоді як порт, що з'єднує два комутатори між собою (або комутатор з маршрутизатором), налаштовують у режимі **trunk port**, здатному одночасно передавати трафік кількох VLAN, розрізняючи їх за допомогою тегу **802.1Q**, чотирибайтної мітки, що вставляється в заголовок Ethernet-кадру й вказує, якому VLAN належить цей конкретний кадр.

Захищеність реалізації VLAN залежить від дисципліни налаштування трункових портів: атака **VLAN hopping**, коротко згадана в підрозділі 13.2, найчастіше експлуатує саме неправильно налаштований порт, що автоматично узгоджує трункінг (deprecated-практика Dynamic Trunking Protocol) замість явного, статичного визначення режиму кожного порту адміністратором. Явне вимкнення автоматичного узгодження трункінгу на всіх портах, крім свідомо призначених для нього, і винесення керуючого (native) VLAN за межі номера, що використовується для реального трафіку, лишаються базовими, але часто ігнорованими практиками безпечного налаштування комутатора.

13.9. Мережева документація: схеми та реєстр IP-адрес (IPAM)

Жодна з розглянутих у цьому розділі технологій сегментації (VLAN, підрозділ 13.8), маршрутизації чи міжмережевого екранування (підрозділ 13.5) не працює безпечно без актуальної документації: адміністратор, який не знає, який VLAN і яка IP-адреса призначені конкретному пристрою, не може ні правильно налаштувати правило доступу, ні швидко локалізувати джерело інциденту під час розслідування, розглянутого в розділі 22. **IPAM (IP Address Management)** позначає клас систем і практик, що ведуть централізований реєстр виданих IP-адрес, підмереж, VLAN-прив'язок і відповідних DNS-записів, замінюючи ненадійні електронні таблиці єдиним джерелом істини (single source of truth) для всієї мережевої інфраструктури.

Актуальна мережева документація, схеми топології та реєстр IPAM набувають особливого практичного значення саме на перетині з моніторингом і

аудитом: SIEM-система (розділ 3 модуля 1), що фіксує підозрілу активність з конкретної IP-адреси, дає значно швидший результат розслідування, якщо ця адреса одразу зіставляється з відомим, задокументованим пристроєм і його власником, а не потребує ручного пошуку. З цієї причини аудит відповідності (розділ 18) регулярно перевіряє саме синхронність між фактичним станом мережі, отриманим інструментами виявлення топології, та задокументованим станом в IPAM, оскільки розбіжність між ними є типовою ознакою тіньової ІТ-інфраструктури (shadow IT) чи забутого, невідстежуваного пристрою.

Практикум 13. Побудова матриці засобів захисту для багаторівневої мережі

Мета роботи: закріпити навички системного аналізу захищеності мережі з використанням моделі OSI як структурної рамки.

Хід роботи:

1. для заданої викладачем типової топології організації (центральний офіс, філія, гостьовий сегмент, сегмент відеоспостереження) скласти перелік мережевого обладнання й протоколів, що використовуються на кожному рівні OSI;
2. для кожного рівня визначити найбільш імовірний вектор атаки з урахуванням специфіки заданої топології;
3. підібрати й обґрунтувати конкретний технічний засіб захисту для кожного визначеного вектора, спираючись на таблицю 13.1;
4. оформити результат у вигляді єдиної матриці «рівень OSI, загроза, засіб захисту, відповідальна роль (з розділу 9)».

Контрольні питання до розділу 13

1. Чому недоцільно аналізувати захищеність мережі без розподілу на рівні моделі OSI?
2. Опишіть механізм атаки ARP spoofing і назвіть засіб протидії на канальному рівні.
3. У чому полягає фізичне розділення площини керування й площини передавання даних у концепції SDN?
4. Які нові ризики для безпеки мережі створює централізація керування в SDN-контролері?
5. Чим відрізняються завдання захисту на транспортному й на прикладному рівнях моделі OSI?

6. Чому недоцільно зосереджувати весь захист мережі лише на одному рівні моделі OSI?
7. Наведіть по одному прикладу типової загрози й засобу захисту для трьох довільних рівнів OSI (таблиця 13.1).
8. Опишіть механізм атаки VLAN hopping і засіб протидії на канальному рівні.
9. Чим статична прив'язка ARP відрізняється від динамічного протоколу ARP за рівнем захищеності?
10. Що таке автентифікація сусідів маршрутизації і від якої загрози вона захищає?
11. Який протокол найпоширеніше використовують для зв'язку між контролером SDN і мережевими пристроями?
12. Назвіть одну перевагу й один новий ризик, які централізація керування в SDN вносить для захищеності мережі.
13. Чим версія TLS 1.3 переважає TLS 1.2 з погляду рекомендацій NIST SP 800-52?
14. Що таке Web Application Firewall (WAF) і на якому рівні моделі OSI він переважно діє?
15. Чим практика Secure SDLC відрізняється від захисту вже розгорнутого застосунку лише засобами WAF?
16. Чому автентифікація і шифрування каналу керування є обов'язковою умовою безпечного впровадження SDN, а не опціональним доповненням?

РОЗДІЛ 14. ХМАРНА ІНФРАСТРУКТУРА: МОДЕЛІ, СПІЛЬНА ВІДПОВІДАЛЬНІСТЬ, БЕЗПЕКА, АДМІНІСТРУВАННЯ



Зміст розділу

14.1. Моделі хмарних послуг: IaaS, PaaS, SaaS

14.2. Моделі розгортання хмари

14.3. Модель спільної відповідальності за безпеку

14.4. Infrastructure as Code: концепція та інструменти

14.5. Відмовостійкість і аварійне відновлення хмарних систем

14.6. Адміністрування хмарних ресурсів

14.7. Захист хмарних сховищ даних

14.8. Безпека безсерверних обчислень (Serverless/FaaS)

14.9. Мережева безпека в хмарі: групи безпеки та віртуальні мережі

14.10. Гібридна зв'язність: з'єднання локальної інфраструктури з хмарою

14.11. FinOps: контроль витрат як складова адміністрування хмари

14.1. Моделі хмарних послуг: IaaS, PaaS, SaaS

Хмарні обчислення (cloud computing) за визначенням NIST SP 800-145 являють собою модель забезпечення повсюдного, зручного мережевого доступу за вимогою до спільного пулу обчислювальних ресурсів (мереж, серверів, сховищ, застосунків, послуг), який можна оперативнo надати або звільнити з мінімальними управлінськими зусиллями. Залежно від того, яку частину стека бере на себе постачальник, а яку лишає споживачу, розрізняють три базові моделі послуг: **IaaS (Infrastructure as a Service)**, коли постачальник надає віртуалізовані обчислювальні ресурси, а споживач самостійно керує операційною системою й вище; **PaaS (Platform as a Service)**, коли постачальник додатково бере на себе керування операційною системою й середовищем виконання, залишаючи споживачу лише розробку й розгортання застосунку; **SaaS (Software as a Service)**, коли споживач отримує готовий до використання застосунок і не керує жодним рівнем інфраструктури під ним.

14.2. Моделі розгортання хмари

Незалежно від моделі послуги хмарну інфраструктуру розгортають за однією з кількох моделей: **публічна хмара** (спільна інфраструктура постачальника, доступна необмеженому колу клієнтів), **приватна хмара** (інфраструктура, виділена для одної організації, розгорнута або в її власному дата-центрі, або як виділений сегмент у постачальника), **гібридна хмара** (поєднання приватної й публічної інфраструктури з можливістю переносити навантаження між ними) та **багатохмарна (multi-cloud) архітектура** (одночасне використання кількох незалежних постачальників публічної хмари, найчастіше для зниження залежності від одного постачальника та підвищення відмовостійкості). Вибір моделі розгортання для конкретної організації визначається не лише вартістю, а й вимогами

регуляторів щодо розташування та обробки даних, що для державних і критичних інфраструктурних систем в Україні часто є вирішальним фактором на користь приватної хмари або суворо контролюваного сегмента гібридної.

14.3. Модель спільної відповідальності за безпеку

Найпоширенішою помилковою уявою серед організацій, що переходять до хмари, є припущення, що постачальник хмарних послуг відповідає за безпеку всього, що працює в його інфраструктурі. Насправді відповідальність розподіляється за **моделлю спільної відповідальності (Shared Responsibility Model)**, межа якої зсувається залежно від обраної моделі послуги, як показано на рисунку 14.1.

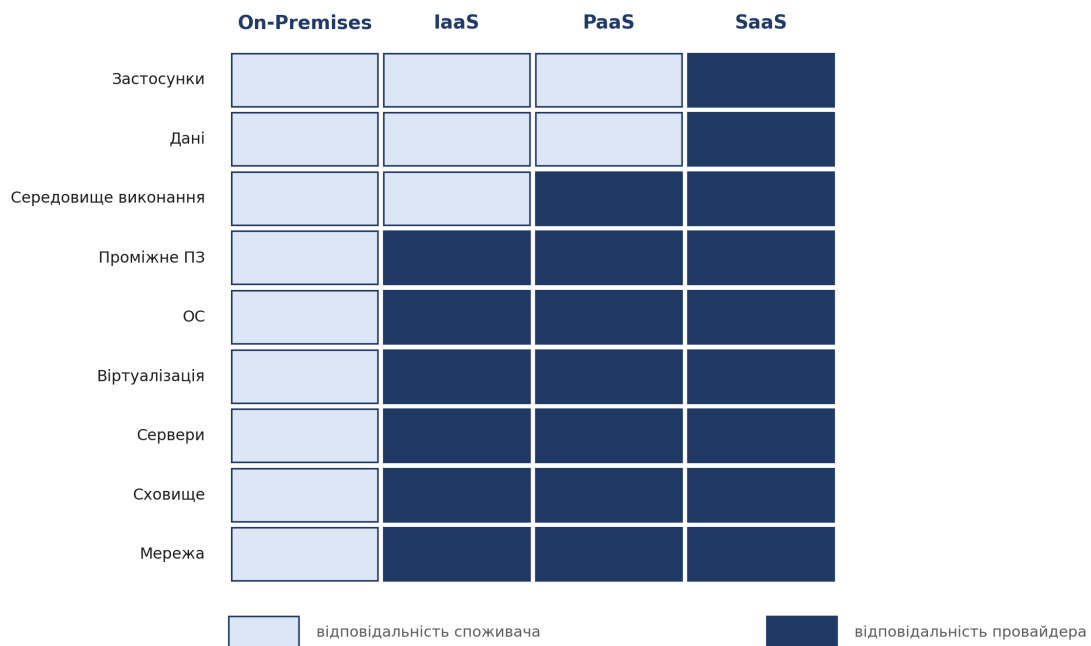


Рис. 14.1. Розподіл відповідальності за безпеку між споживачем і провайдером залежно від моделі хмарної послуги

Джерело: розроблено автором

У моделі IaaS постачальник відповідає лише за фізичну інфраструктуру, мережу та шар віртуалізації, тоді як усе, починаючи з операційної системи (оновлення, налаштування, антивірусний захист) і вище, лишається на споживачеві. У моделі SaaS, навпаки, постачальник відповідає за весь стек аж до застосунку

включно, а споживач відповідає практично лише за коректне налаштування прав доступу користувачів і класифікацію власних даних. Практичний висновок для адміністратора безпеки: перш ніж підписувати угоду з хмарним постачальником, необхідно письмово з'ясувати, де саме проходить межа відповідальності для обраної послуги, оскільки типовою причиною реальних інцидентів у хмарі є не злом інфраструктури постачальника, а неправильно налаштований компонент на боці споживача, найчастіше публічно доступне сховище даних або надмірно широкі права доступу.

14.4. Infrastructure as Code: концепція та інструменти

Infrastructure as Code (IaC) являє собою підхід, за якого конфігурація хмарної (і не лише хмарної) інфраструктури описується у вигляді декларативних файлів, а не встановлюється вручну через консоль керування. Файли опису зберігаються в системі контролю версій (найчастіше Git) поряд із програмним кодом застосунку, що дозволяє застосувати до інфраструктури ті самі практики, що й до звичайного коду: рецензування змін, історію версій, автоматизоване розгортання. Рисунок 14.2 показує типовий конвеєр IaC та його застосування для автоматизованого відновлення інфраструктури після збою.



*Рис. 14.2. Конвеєр Infrastructure as Code та автоматизоване відновлення
хмарної інфраструктури після збою
Джерело: розроблено автором*

Серед поширених інструментів ІаС розрізняють декларативні системи оркестрації розгортання (Terraform, який завдяки хмаронезалежному підходу дозволяє описувати ресурси кількох постачальників в одному проєкті) та системи управління конфігурацією (Ansible, Puppet, Chef, вже згадані в розділі 11 стосовно групових політик). Ключова перевага підходу ІаС для захищеності системи, а не лише для зручності розгортання, розкривається саме у застосуванні до аварійного відновлення: якщо повний опис інфраструктури зберігається як код, втрачений унаслідок збою чи навіть цілеспрямованої атаки вузол чи навіть цілий сегмент хмарної інфраструктури можна відтворити автоматизованим розгортанням з того самого опису, без ручного, тривалого й схильного до помилок відновлення налаштувань за пам'яттю адміністратора. Саме цей метод автоматизованого відновлення хмарної системи на основі концепції Infrastructure as Code докладно розглянуто в [20] зі списку джерел цього розділу.

14.5. Відмовостійкість і аварійне відновлення хмарних систем

Показники **RPO** і **RTO**, вже введені в розділі 11 стосовно базових служб, набувають особливого значення для хмарної інфраструктури, оскільки хмарні платформи надають готові до використання механізми, що дозволяють суттєво наблизити ці показники до нуля, якщо адміністратор свідомо їх налаштував: реплікацію даних сховища між географічно розділеними регіонами постачальника, автоматичне перемикання на резервний регіон (failover) у разі недоступності основного, регулярні автоматизовані знімки стану (snapshots) віртуальних машин і баз даних.

Водночас наявність цих механізмів у постачальника не звільняє організацію від відповідальності за власний план аварійного відновлення (Disaster Recovery Plan): відповідно до моделі спільної відповідальності з підрозділу 14.3, саме споживач визначає, які саме дані й сервіси критичні, які значення RPO/RTO для них прийнятні, і саме споживач відповідає за налаштування конкретних механізмів резервування, а не постачальник за замовчуванням. Практика регулярного тестування процедури відновлення (а не лише наявність налаштованого резервного копіювання) лишається такою самою обов'язковою вимогою в хмарі, якою вона була для локальної інфраструктури в розділі 11.

14.6. Адміністрування хмарних ресурсів

Адміністрування хмарної інфраструктури має кілька специфічних рис порівняно з локальною ІТС. **Керування ідентичністю та доступом (Identity and Access Management, IAM)** у хмарі побудоване на детальних, гранульованих політиках дозволів, що визначають не просто «адміністратор чи користувач», а точний перелік дозволених дій над конкретними ресурсами; принцип найменших привілеїв із розділу 9 у хмарному середовищі є не побажанням, а технічно обов'язковою практикою, оскільки надмірно широка IAM-політика є однією з найпоширеніших причин реальних витоків даних із хмарних сховищ. **Моніторинг і контроль витрат (cost management)** стає окремою адміністративною функцією, відсутньою в локальній інфраструктурі: оскільки хмарні ресурси тарифікуються за фактичне використання, помилково залишений або скомпрометований і використовуваний зловмисником для майнінгу ресурс може за лічені дні призвести до непропорційно високих витрат. **Тегування ресурсів** (позначення кожного ресурсу метаданими про власника, призначення, середовище) є базовою гігієною хмарного адміністрування, без якої за кілька місяців експлуатації неможливо визначити призначення й відповідальну особу для більшості наявних ресурсів.

14.7. Захист хмарних сховищ даних

Підрозділ 14.3 назвав публічно доступне сховище даних найпоширенішою практичною причиною реальних витоків у хмарі; розглянемо, чому саме ця помилка трапляється так часто. Об'єктні сховища хмарних постачальників (Amazon S3, Azure Blob Storage, Google Cloud Storage) за задумом підтримують гнучке налаштування прав доступу, включно з можливістю зробити вміст сховища доступним анонімно через посилання, що зручно для розповсюдження публічного контенту (зображень сайту, дистрибутивів програмного забезпечення), але катастрофічне за помилкового застосування до сховища з чутливими даними. Оскільки таке налаштування є звичайною, а не аварійною конфігурацією служби, традиційні засоби виявлення вторгнень (розділ 12), орієнтовані на аномальну поведінку, найчастіше взагалі не фіксують подібну помилку: доступ до неправильно відкритого сховища зовні виглядає як цілком легітимний запит.

Для системного виявлення таких помилок конфігурації, а не лише сховищ, а й ширшого кола хмарних ресурсів (IAM-політик, мережеских груп безпеки, налаштувань шифрування), застосовують клас засобів **CSPM (Cloud Security Posture Management)**: автоматизоване, безперервне сканування всієї хмарної

інфраструктури на відповідність базовим правилам захищеності (наприклад, «жодне сховище не повинно бути доступне анонімно», «шифрування в стані спокою повинно бути увімкнене для всіх баз даних») із формуванням списку виявлених відхилень для адміністратора, за принципом, аналогічним скануванню вразливостей із підрозділу 9.6, але спрямованим саме на помилки конфігурації, а не на відомі вразливості програмного забезпечення.

14.8. Безпека безсерверних обчислень (Serverless/FaaS)

Модель **FaaS (Function as a Service)**, різновид безсерверних (serverless) обчислень, доводить логіку підрозділу 14.1 до крайньої точки: розробник передає постачальнику не лише операційну систему й середовище виконання, як у PaaS, а й саме управління обчислювальним ресурсом як таким, розгортаючи лише окрему функцію, яка виконується у відповідь на подію (HTTP-запит, повідомлення черги, зміна файлу у сховищі) і автоматично звільняється одразу після завершення. Межа моделі спільної відповідальності (підрозділ 14.3) для FaaS зсувається ще далі на користь постачальника: споживач відповідає практично виключно за безпеку коду власної функції та за дозволи, надані їй.

Саме останнє є характерним джерелом ризику: оскільки кожна функція за своєю природою маленька й вузькоспеціалізована, організації, що активно використовують FaaS, часто нараховують сотні окремих функцій, кожна з яких потребує власного, ретельно обмеженого набору дозволів IAM (підрозділ 14.6). Поширеною помилкою є надання всім функціям одного надлишково широкого набору дозволів «про запас», що прямо суперечить принципу найменших привілеїв (підрозділ 9.4) і перетворює компрометацію єдиної, навіть незначної функції на потенційний доступ до значно ширшого кола ресурсів, ніж ця функція фактично потребує для роботи.

14.9. Мережева безпека в хмарі: групи безпеки та віртуальні мережі

Сегментація й міжмережеві екрани, розглянуті для локальної інфраструктури в підрозділах 10.4 і 13.8, мають власний, специфічний для хмари відповідник. Замість фізичного чи VLAN-розмежування хмарні платформи оперують поняттям **віртуальної мережі (Virtual Network, VNet в Azure, VPC в AWS)**: логічно ізольованого мережевого простору всередині хмарної інфраструктури постачальника, поділеного на підмережі за тим самим принципом, що вводився для локальної мережі. Фільтрація трафіку між ресурсами всередині віртуальної мережі

й ззовні реалізується через **групи безпеки (Security Groups, Network Security Groups)**: набори правил дозволу й заборони трафіку, що прикріплюються не до фізичного порту комутатора, як у розділі 10, а безпосередньо до віртуального мережевого інтерфейсу конкретного ресурсу, що дає змогу застосовувати політику мікросегментації (підрозділ 13.6) навіть без окремого контролера SDN, оскільки сама хмарна платформа вже виконує цю роль.

Практична відмінність від локального адміністрування полягає в тому, що групи безпеки визначаються декларативно, тим самим підходом Infrastructure as Code, що вводився в підрозділі 14.4, а отже підлягають рецензуванню змін і контролю версій так само, як і будь-який інший програмний код, а не лише вручну редагуються через консоль керування, як типова конфігурація локального міжмережевого екрана.

14.10. Гібридна зв'язність: з'єднання локальної інфраструктури з хмарою

Підрозділ 14.2 згадав гібридну хмару як модель розгортання, що поєднує локальну й хмарну інфраструктуру; розглянемо, як технічно реалізується мережеве з'єднання між ними. Найпростіший варіант, **VPN через інтернет** (той самий site-to-site VPN, розглянутий у підрозділі 10.6, але тепер один з кінців тунелю розташований у хмарі, а не на іншому локальному майданчику організації), не вимагає додаткової фізичної інфраструктури, але успадковує змінну затримку й пропускну здатність публічного інтернету. Для навантажень, чутливих до затримки чи обсягу переданих даних, хмарні провайдери пропонують **виділене приватне з'єднання** (ExpressRoute в Azure, Direct Connect в AWS): фізичний канал через партнера-оператора зв'язку, що з'єднує локальну мережу організації безпосередньо з мережею постачальника хмари, минаючи публічний інтернет повністю.

Крім передбачуваної пропускну здатності й затримки, виділене з'єднання має й переваги безпеки: трафік між локальною інфраструктурою і хмарою не проходить публічним інтернетом, а отже не наражається на ту саму поверхню атаки, що трафік VPN-тунелю, хоча сам факт приватності каналу не звільняє адміністратора від застосування груп безпеки (підрозділ 14.9) і принципу найменших привілеїв до трафіку, що цим каналом проходить: приватність каналу зменшує один клас ризиків, але не замінює решту заходів захисту, розглянутих у цьому розділі.

14.11. FinOps: контроль витрат як складова адміністрування хмари

Модель спільної відповідальності (підрозділ 14.3) і еластичність хмарних ресурсів, розглянуті раніше в цьому розділі, мають зворотний бік: помилково залишений увімкненим тестовий ресурс, неоптимальний вибір типу віртуальної машини чи забутий знімок диска перетворюються не на проблему доступності, а безпосередньо на фінансові витрати, що накопичуються автоматично й непомітно. **FinOps** позначає дисципліну й культуру управління хмарними витратами, що об'єднує інженерів, фінансовий та управлінський персонал довкола спільної відповідальності за економічну ефективність хмарної інфраструктури, аналогічно до того, як DevSecOps (підрозділ 9.9 модуля 2) об'єднує розробку й безпеку.

З погляду адміністратора, практичний бік FinOps спирається на ті самі принципи маркування ресурсів (tagging) і моніторингу, що вже розглядалися для безпеки (підрозділ 14.6): послідовне маркування кожного ресурсу власником, проектом і середовищем (production/staging/test) дозволяє автоматично зіставляти витрати з конкретною командою чи сервісом і виявляти аномальне зростання витрат так само, як SIEM виявляє аномальну мережеву активність. Автоматичне вимкнення чи видалення невикористовуваних ресурсів за розкладом, а також бюджетні сповіщення (budget alerts), що спрацьовують при наближенні витрат до заданого порогу, є прямим застосуванням у фінансовій площині того самого підходу проактивного моніторингу з порогоми й сповіщеннями, що детально розглянуто в розділі 2 модуля 1.

Практикум 14. Опис навчального хмарного середовища у форматі Infrastructure as Code

Мета роботи: отримати практичні навички декларативного опису хмарної інфраструктури та оцінити відмінність цього підходу від ручного налаштування через консоль керування.

Хід роботи:

1. обрати навчальний обліковий запис одного з публічних хмарних постачальників (з використанням безкоштовного рівня використання) або локальний емулятор;
2. вручну через консоль керування розгорнути просту інфраструктуру (одна віртуальна машина, одна мережа, одна група безпеки) і зафіксувати час, витрачений на кожен крок;
3. описати ідентичну за результатом інфраструктуру декларативним файлом (наприклад, засобами Terraform) і розгорнути її автоматизовано;
4. навмисно видалити один ресурс і повторно застосувати опис IaC, підтвердивши автоматичне відновлення попереднього стану;
5. порівняти обидва підходи за часом, відтворюваністю результату та зручністю внесення змін, оформивши висновок у звіті.

Примітка для викладача. Конкретний перелік послуг, безкоштовних лімітів і актуальний синтаксис інструменту IaC необхідно звіряти з чинною документацією обраного постачальника на момент проведення заняття, оскільки хмарні платформи змінюються значно швидше за друковані джерела.

Контрольні питання до розділу 14

1. Чим принципово відрізняються моделі послуг IaaS, PaaS і SaaS з погляду обсягу керування, що лишається на споживачеві?
2. Поясніть на власному прикладі, як модель спільної відповідальності визначає, хто відповідає за неправильно налаштоване хмарне сховище даних.
3. У чому полягає підхід Infrastructure as Code і чим він відрізняється від ручного налаштування через консоль керування?
4. Як опис інфраструктури у вигляді коду прискорює аварійне відновлення після втрати вузла чи сегмента?
5. Назвіть специфічні для хмарного середовища функції адміністрування, яких не існує (або які виражені значно слабше) в локальній інфраструктурі.

6. Дайте означення моделей IaaS, PaaS, SaaS і наведіть по одному прикладу типового продукту для кожної.
7. Чим приватна хмара відрізняється від гібридної за архітектурою розгортання?
8. Навіщо організації можуть свідомо обирати багатохмарну (multi-cloud) архітектуру?
9. Поясніть межу відповідальності за безпеку в моделі IaaS: що саме лишається на споживачеві (рисунок 14.1)?
10. Яка типова причина реальних інцидентів у хмарі, згідно з підрозділом 14.3?
11. Чим Terraform як інструмент оркестрації відрізняється від Ansible як системи управління конфігурацією?
12. Дайте означення показників RPO і RTO стосовно хмарної інфраструктури.
13. Чому наявність механізмів резервування в постачальника не звільняє організацію від власного плану аварійного відновлення?
14. Що таке тегування ресурсів (resource tagging) і яку практичну проблему воно вирішує в хмарному адмініструванні?
15. Чому надмірно широка IAM-політика вважається однією з найпоширеніших причин витоків даних із хмарних сховищ?
16. Які специфічні для хмари ризики пов'язані з помилково залишеним або скомпрометованим ресурсом і моделлю тарифікації за використання?
17. Чому регуляторні вимоги щодо розташування даних можуть переважити чисто економічну вигоду публічної хмари для державних систем?

РОЗДІЛ 15. КОНТЕЙНЕРИЗАЦІЯ ТА KUBERNETES: БЕЗПЕКА І АДМІНІСТРУВАННЯ



Зміст розділу

15.1. Контейнери проти віртуальних машин

15.2. Docker: архітектура та основні поняття

15.3. Архітектура кластера Kubernetes

15.4. Безперервна інтеграція та доставка контейнеризованих сервісів

15.5. Безпека контейнерів і Kubernetes

15.6. Секрети та конфігурація в Kubernetes

15.7. Service Mesh: взаємна автентифікація та безпека трафіку між мікросервісами

15.8. Спостережуваність контейнеризованих застосунків

15.9. Управління ресурсами та автоматичне масштабування

15.10. GitOps: декларативне управління конфігурацією Kubernetes

15.1. Контейнери проти віртуальних машин

Контейнеризація є ще одним, вужчим за віртуалізацію, рівнем ізоляції застосунків. Головна архітектурна відмінність контейнера від віртуальної машини показана на рисунку 15.1: віртуальна машина віртуалізує апаратне забезпечення й запускає для кожного застосунку повноцінну гостьову операційну систему з власним ядром, тоді як контейнери спільно використовують одне ядро хостової операційної системи, а ізолюються засобами самого ядра (просторами імен і контрольними групами процесів у Linux).



*Рис. 15.1. Порівняння архітектури віртуальних машин і контейнерів
Джерело: розроблено автором*

Наслідком цієї відмінності є суттєво менший розмір і час запуску контейнера порівняно з віртуальною машиною (секунди чи навіть частки секунди проти хвилин), що робить контейнери природним будівельним блоком для архітектури мікросервісів, коли застосунок свідомо розбивають на багато незалежно розгортаних і масштабованих компонентів. Зворотним боком спільного використання ядра хостової ОС є ширша поверхня потенційної атаки в разі вразливості самого ядра чи неправильної конфігурації ізоляції: питання безпеки контейнерів докладно розглянуто в підрозділі 15.5.

15.2. Docker: архітектура та основні поняття

Docker лишається найпоширенішим інструментом створення й запуску контейнерів, хоча специфікація формату контейнерних образів (OCI, Open Container Initiative) стандартизована й підтримується кількома сумісними рушіями. Базовими поняттями екосистеми Docker є: **образ (image)**, незмінний шаблон файлової системи та метаданих застосунку, що описується декларативним файлом Dockerfile; **контейнер (container)**, запущений і, на відміну від образу, змінний під час роботи екземпляр образу; **реєстр (registry)**, сховище образів, готових для завантаження й розгортання (Docker Hub як публічний приклад або приватний реєстр організації).

Практика побудови образу за принципом **багатоетапного складання (multi-stage build)**, коли інструменти й залежності, потрібні лише для компіляції застосунку, не потрапляють у фінальний образ, є визнаним стандартом як для зменшення розміру готового образу, так і, що важливіше для захищеності, для зменшення кількості зайвого програмного забезпечення, а отже й потенційних вразливостей, у фінальному образі, що розгортається в продуктивному середовищі.

15.3. Архітектура кластера Kubernetes

Один-єдиний хост Docker добре підходить для розробки й невеликих навантажень, але керування сотнями контейнерів на десятках серверів, з автоматичним перезапуском відмовних екземплярів, масштабуванням і розподілом навантаження, потребує окремого рівня оркестрації. **Kubernetes** є найпоширенішою системою оркестрації контейнерів. Рисунок 15.2 зображує основні компоненти кластера Kubernetes.

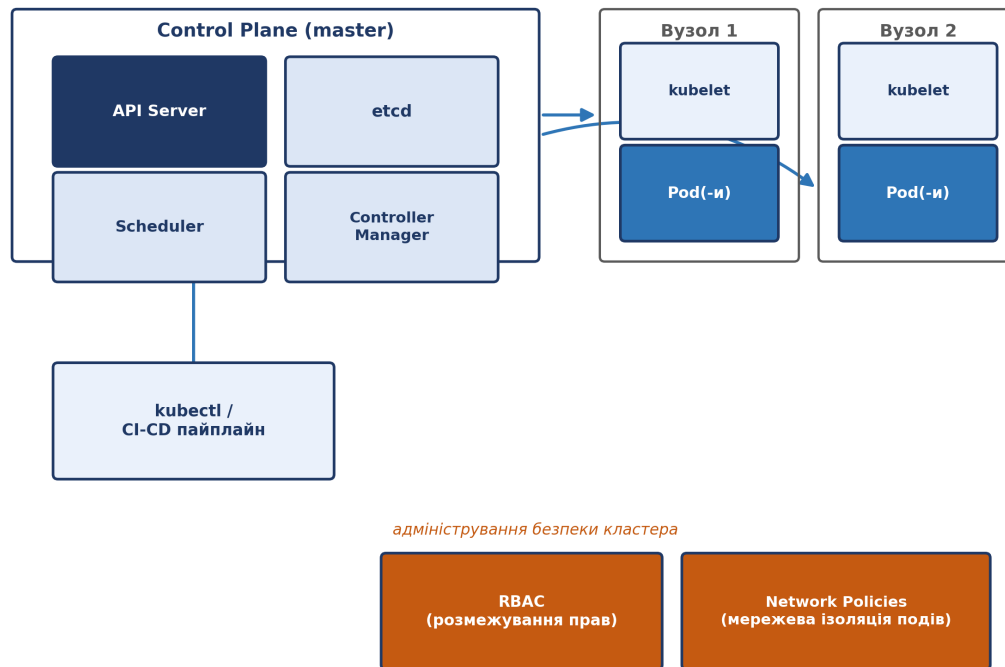


Рис. 15.2. Архітектура кластера Kubernetes: control plane та робочі вузли
Джерело: розроблено автором

Control plane (керівна площина) кластера складається з кількох компонентів: **API Server**, єдина точка входу для всіх команд керування кластером (зокрема, від утиліти командного рядка `kubectl`); **etcd**, розподілена база даних типу «ключ-значення», у якій зберігається весь стан кластера; **Scheduler**, компонент, що вирішує, на якому робочому вузлі розмістити новий под; **Controller Manager**, що безперервно звіряє фактичний стан кластера з бажаним і за потреби ініціює коригувальні дії, наприклад перезапуск відмовного поду. На кожному **робочому вузлі** працює агент **kubelet**, що виконує інструкції control plane і підтримує задані **поди** (pod), найменшу одиницю розгортання в Kubernetes, яка об'єднує один або кілька тісно пов'язаних контейнерів. Показане на рисунку 15.2 адміністрування безпеки кластера (RBAC, мережеві політики) докладно розглянуто в підрозділі 15.5.

15.4. Безперервна інтеграція та доставка контейнеризованих сервісів

Контейнеризація природно поєднується з практикою **безперервної інтеграції (Continuous Integration, CI)**: оскільки контейнерний образ є відтворюваним, самодостатнім артефактом, конвеєр CI може автоматично зібрати образ із щойно

зміненого коду, прогнати автоматизовані тести в ізолюваному середовищі, перевірити якість коду й лише після успішного проходження всіх етапів опублікувати новий образ у реєстрі, звідки він (за наявності конвеєра безперервної доставки, Continuous Delivery) автоматично розгортається в кластері Kubernetes.

Типовий конвеєр такого роду поєднує систему контролю версій (найчастіше Git), сервер автоматизації складання, інструмент статичного аналізу якості коду та засоби контейнеризації й доставки; практична реалізація подібного конвеєра на основі вільного програмного забезпечення (GitLab як система контролю версій, Jenkins як сервер автоматизації, SonarQube для аналізу якості коду, Docker для розгортання застосунку та Nexus як локальний реєстр) детально розглянута в [19] зі списку джерел цього розділу. Ключовий висновок для адміністратора захищеної ІТС полягає в тому, що саме на цьому етапі, до потрапляння образу в продуктивне середовище, найдешевше й найефективніше інтегрувати автоматизовану перевірку безпеки: аналіз коду на вразливості та сканування самого зібраного образу, розглянуте в підрозділі 15.5, а не покладатися виключно на захист уже запущеного контейнера.

15.5. Безпека контейнерів і Kubernetes

Захист контейнеризованого середовища вимагає заходів на кількох рівнях одночасно. **Сканування образів на вразливості** (наприклад, засобами Trivy, Clair чи вбудованими можливостями реєстру) виявляє відомі вразливості в бібліотеках, з яких зібрано образ, до моменту його розгортання, і є природним продовженням конвеєра CI з підрозділу 15.4. **RBAC (Role-Based Access Control)** у Kubernetes розмежовує права доступу до ресурсів кластера за ролями, реалізуючи для контейнерного середовища той самий принцип найменших привілеїв, що вводився в розділі 9: розробник, якому потрібен доступ на перегляд журналів одного простору імен, не повинен мати прав на видалення ресурсів усього кластера.

Мережеві політики (Network Policies) обмежують дозволену мережеву взаємодію між подами, реалізуючи для кластера Kubernetes той самий принцип мікросегментації, що застосовувався до традиційної мережі в розділі 10: за замовчуванням Kubernetes дозволяє подам вільно звертатися один до одного в межах кластера, і саме явно визначені мережеві політики обмежують це до дійсно необхідної комунікації. Додатковими заходами є заборона запуску контейнера від імені користувача root усередині образу, використання незмінної (read-only) файлової системи контейнера там, де це можливо, і регулярне оновлення базових

образів, оскільки застарілий базовий образ є одним із найпоширеніших джерел відомих вразливостей у продуктивному кластері.

15.6. Секрети та конфігурація в Kubernetes

Контейнеризований застосунок, як і будь-який інший, потребує доступу до чутливих даних для власної роботи: паролів бази даних, ключів API, сертифікатів. Kubernetes надає для цього окремий об'єкт **Secret**, відмінний від об'єкта **ConfigMap**, призначеного для нечутливих параметрів конфігурації (адреси серверів, прапорці режиму роботи): обидва об'єкти технічно передають дані в под подібним чином, але лише Secret передбачений саме для чутливої інформації. Водночас варто чітко розуміти обмеження механізму за замовчуванням: без додаткового налаштування вміст Secret зберігається в etcd (підрозділ 15.3) лише в кодуванні base64, а не в криптографічно захищеному вигляді, що є поширеним і небезпечним хибним уявленням серед початківців.

Зрілі кластери тому доповнюють вбудований механізм Secret або увімкненням шифрування вмісту etcd у стані спокою, або, частіше, інтеграцією з зовнішнім сховищем секретів на кшталт того, що вже згадувалося як компонент рішень РАМ у підрозділі 10.7, коли под отримує чутливе значення динамічно під час запуску, а не зберігає його постійно в об'єкті Kubernetes. Ротація секретів (регулярна заміна паролів і ключів на нові) для контейнеризованих застосунків так само лишається обов'язковою практикою, як і для традиційної інфраструктури.

15.7. Service Mesh: взаємна автентифікація та безпека трафіку між мікросервісами

Архітектура мікросервісів, згадана в підрозділі 15.1 як природний наслідок легкості контейнерів, породжує специфічну проблему масштабу: сотні мікросервісів, що активно обмінюються мережевими запитами один з одним усередині кластера, важко захистити й спостерігати індивідуальним налаштуванням кожного сервісу. Клас рішень **Service Mesh** (наприклад Istio, Linkerd) вирішує цю проблему, впроваджуючи в кожен под допоміжний контейнер-посередник (sidecar проху), що прозоро для застосунку перехоплює весь вхідний і вихідний мережевий трафік поду.

Із погляду захищеності найважливішою можливістю service mesh є автоматичне забезпечення **взаємного TLS (mutual TLS, mTLS)** між усіма сервісами кластера: на відміну від звичайного TLS, розглянутого в підрозділі 13.5, де лише

сервер пред'являє сертифікат клієнту, за взаємного TLS обидві сторони з'єднання пред'являють і перевіряють сертифікати одна одної, тому жоден под не може видати себе за інший, легітимний сервіс усередині кластера. Оскільки *sidecar proxy* впроваджується прозоро, розробникам мікросервісів не потрібно окремо реалізовувати шифрування й автентифікацію трафіку в коді кожного сервісу: ця відповідальність повністю виноситься на рівень інфраструктури, за тим самим принципом розділення відповідальності, що спостерігався в моделі спільної відповідальності хмари (підрозділ 14.3).

15.8. Спостережуваність контейнеризованих застосунків

Централізоване журналювання, розглянуте в підрозділі 2.7 стосовно контейнерів, покриває лише одну складову повноцінної **спостережуваності (observability)** кластера Kubernetes: журнали (logs) відповідають на запитання «що сталося», але не завжди ефективні для запитання «наскільки добре зараз працює система». Другою складовою є **метрики (metrics)**: числові часові ряди про використання ресурсів кожним подом і вузлом (завантаження процесора, використання пам'яті, кількість запитів на секунду), для збору й зберігання яких у хмарному й контейнеризованому середовищі де-факто стандартом є **Prometheus**, система моніторингу з відкритим кодом, що періодично опитує (принцип, споріднений із SNMP-моніторингом з підрозділу 1.5 модуля 1) кожен компонент кластера через спеціальний ендпоінт, і **Grafana** для побудови дашбордів на основі цих метрик, функціонально паралельних до Workbook у Sentinel (підрозділ 4.8 модуля 1).

Третьою, найскладнішою для впровадження складовою є **розподілене трасування (distributed tracing)**: у мікросервісній архітектурі (підрозділ 15.1) один запит користувача часто проходить через десяток окремих сервісів, і за відсутності трасування з'ясувати, який саме з них спричинив уповільнення чи помилку, вкрай важко. Трасування призначає кожному вхідному запиту унікальний ідентифікатор, що передається між усіма сервісами, залученими до обробки цього запиту (природно, через той самий *sidecar proxy service mesh*, підрозділ 15.7), дозволяючи згодом відтворити повний шлях запиту й час, витрачений у кожному сервісі, аналогічно до того, як ланцюг зберігання доказів (підрозділ 2.10 модуля 1) відтворює шлях доказу, тільки для запиту користувача, а не для об'єкта розслідування.

15.9. Управління ресурсами та автоматичне масштабування

Под, розгорнутий без явно заданих обмежень споживання ресурсів, становить прихований операційний ризик: один процес, що вийшов з-під контролю (наприклад, через витік пам'яті в коді застосунку), здатний вичерпати ресурси всього робочого вузла (підрозділ 15.3) і тим самим вплинути на всі інші поди, розміщені на цьому вузлі, а не лише на себе. Тому для кожного контейнера рекомендується явно задавати **запит ресурсів (resource request)**, гарантований мінімум, який планувальник (Scheduler) враховує при виборі вузла для розміщення поду, і **ліміт ресурсів (resource limit)**, максимум, який под не може перевищити: перевищення ліміту пам'яті призводить до примусового завершення контейнера, а перевищення ліміту процесора, до обмеження (throttling) виконання.

На основі цих же метрик споживання (підрозділ 15.8) Kubernetes здатний автоматично масштабувати навантаження: **Horizontal Pod Autoscaler (HPA)** збільшує чи зменшує кількість запущених копій поду залежно від поточного навантаження (наприклад, утримуючи середнє використання процесора на рівні 70%), тоді як **Cluster Autoscaler** йде на рівень вище й змінює саму кількість робочих вузлів кластера, коли наявних вузлів уже недостатньо для розміщення потрібної кількості подів. Для захищеної ІТС автоматичне масштабування має й аспект стійкості до навантажувальних атак: правильно налаштований HPA дозволяє кластеру частково пом'якшити наслідки об'ємної DDoS-атаки (підрозділ 6.6 модуля 1) на застосунок прикладного рівня, хоча й не замінює спеціалізовані засоби захисту від DDoS на рівні мережі.

15.10. GitOps: декларативне управління конфігурацією Kubernetes

Мережева політика й маніфести розгортання, розглянуті в попередніх підрозділах, зазвичай описуються декларативно у форматі YAML; природним наступним кроком є зберігання цих файлів не локально на робочій станції адміністратора, а в системі контролю версій (Git), звідки їх автоматично й відтворювано застосовує до кластера спеціалізований інструмент. **GitOps** позначає саме таку практику: Git-репозиторій стає єдиним джерелом істини (single source of truth) для бажаного стану кластера, а інструменти на кшталт **ArgoCD** чи **Flux** безперервно порівнюють фактичний стан кластера із задекларованим у репозиторії й автоматично усувають будь-яке розходження, у тому числі несанкціоновану ручну зміну, внесену напрямку через kubectl в обхід процесу перегляду.

Безпековий ефект GitOps прямо перетинається з темами, вже розглянутими в цьому й попередніх розділах: кожна зміна конфігурації проходить той самий процес перегляду коду (code review) і затвердження (pull request), що й звичайний прикладний код, забезпечуючи повний, незмінний аудиторський слід (audit trail) у самій історії Git-репозиторію, замість розрізнених записів у журналах адміністрування (підрозділ 21 модуля 3). Такий підхід є практичною реалізацією принципу Infrastructure as Code, згаданого в підрозділі 9.9 модуля 2, поширеною тепер не лише на створення інфраструктури, а й на постійне підтримання її захищеного, узгодженого стану.

Практикум 15. Розгортання застосунку в Kubernetes із застосуванням RBAC та мережевої політики

Мета роботи: отримати практичні навички розгортання контейнеризованого застосунку в кластері Kubernetes та базового адміністрування його безпеки.

Хід роботи:

1. розгорнути навчальний кластер Kubernetes (наприклад, засобами Minikube або kind, локально) і переконатися в доступності control plane через kubectl;
2. описати й розгорнути простий под із контейнером на основі попередньо зібраного навчального образу, перевіривши попередньо цей образ засобом сканування вразливостей;
3. створити окрему службову роль (Role) з обмеженими правами лише на перегляд ресурсів одного простору імен та прив'язати її до тестового облікового запису служби (ServiceAccount), підтвердивши, що дії поза межами наданих прав відхиляються кластером;
4. створити мережеву політику, що дозволяє вхідний трафік до розгорнутого поду лише з визначеного іншого поду, і перевірити її дію спробою звернення з дозволеного та з недозволеного джерела;
5. оформити звіт із висновком про те, які з застосованих налаштувань відповідають принципу найменших привілеїв із розділу 9.

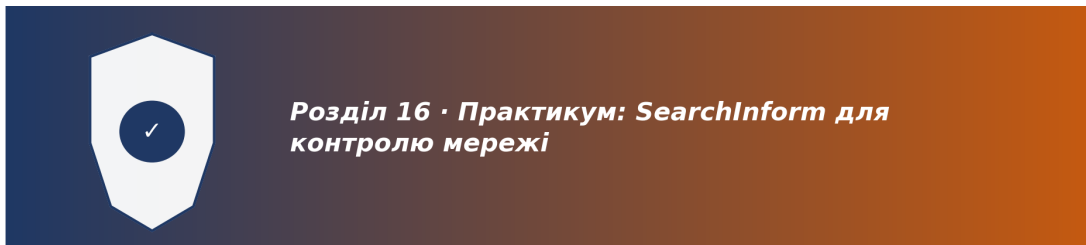
Примітка для викладача. Конкретні команди й синтаксис маніфестів необхідно звіряти з чинною документацією Kubernetes на момент проведення заняття, оскільки API кластера версіюється й окремі можливості можуть змінювати статус (alpha/beta/stable) між релізами.

Контрольні питання до розділу 15

1. У чому головна архітектурна відмінність контейнера від віртуальної машини та які наслідки це має для швидкості розгортання?
2. Що таке багатоетапне складання образу Docker і чому воно корисне з погляду безпеки, а не лише розміру образу?
3. Перелічіть основні компоненти control plane Kubernetes і коротко опишіть роль кожного.
4. Як конвеєр CI/CD для контейнеризованого застосунку дозволяє виявляти проблеми безпеки до потрапляння коду в продуктивне середовище?

5. Поясніть, як RBAC і мережеві політики Kubernetes реалізують для кластера ті самі принципи, що розглядалися в розділах 9 і 10 для традиційної ІТС.
6. Чому контейнери запускаються значно швидше за віртуальні машини?
7. Що таке образ (image) і чим він відрізняється від запущеного контейнера?
8. Яку роль виконує API Server у control plane Kubernetes (рисунок 15.2)?
9. Для чого потрібен компонент etcd у кластері Kubernetes?
10. Чим Scheduler відрізняється від Controller Manager за функцією?
11. Що таке под (pod) і чому він, а не окремий контейнер, є найменшою одиницею розгортання в Kubernetes?
12. Опишіть типовий конвеєр CI/CD для контейнеризованого застосунку: з яких компонентів він складається?
13. Чому найдешевше й найефективніше інтегрувати перевірку безпеки саме на етапі CI, а не після розгортання застосунку?
14. Що таке RBAC у Kubernetes і який загальний принцип адміністрування безпеки він реалізує для кластера?
15. Яку задачу вирішують мережеві політики (Network Policies) у Kubernetes за замовчуванням і після налаштування?
16. Назвіть щонайменше три додаткові заходи безпеки контейнерів, окрім RBAC і мережевих політик.
17. Що таке сканування образів на вразливості і на якому етапі конвеєра його найдоцільніше виконувати?
18. Чому заборона запуску контейнера від імені root усередині образу вважається базовою практикою безпеки?

РОЗДІЛ 16. ПРАКТИКУМ: НАЛАШТУВАННЯ SEARCHINFORM ДЛЯ КОНТРОЛЮ МЕРЕЖІ



Зміст розділу

- 16.1. Призначення та архітектура SearchInform*
- 16.2. Налаштування політики контролю мережевих каналів*
- 16.3. Розслідування інцидентів за допомогою повнотекстового індексу*
- 16.4. Інтеграція SearchInform з іншими засобами моніторингу організації*
- 16.5. Продуктивність і масштабування SearchInform у великих організаціях*
- 16.6. Типові причини невдалого впровадження DLP*
- 16.7. Огляд альтернативних рішень на ринку DLP*
- 16.8. Юридичний супровід впровадження DLP: узгодження з профспілкою/трудовим колективом*

16.1. Призначення та архітектура SearchInform

Розділ 7 модуля 1 розглядав DLP-системи загалом; цей розділ, відповідно до практикуму робочої програми, зосереджується на практичному налаштуванні конкретного поширеного в україномовному сегменті рішення класу DLP, **SearchInform**, з погляду саме адміністратора, який виконує первинне розгортання й налаштування контролю мережевих каналів. Архітектурно SearchInform побудований за класичною для DLP-систем схемою «агент-сервер», зображеною на рисунку 16.1: на кожній контрольованій робочій станції встановлюється агент-перехоплювач, що передає дані на центральний сервер для індексування й аналізу.



*Рис. 16.1. Архітектура «агент-сервер» системи SearchInform
Джерело: розроблено автором*

Зібрані агентами дані потрапляють до централізованого повнотекстового індексу, що дозволяє офіцеру інформаційної безпеки виконувати ретроспективний пошук за змістом (а не лише за метаданими) серед усього перехопленого трафіку: електронного листування, повідомлень месенджерів, файлів, переданих через контрольовані канали. Адміністрування такої системи охоплює розгортання й підтримання агентів на робочих станціях, забезпечення продуктивності та дискового простору серверної частини під зростання індексу, а також, що є предметом наступного підрозділу, безпосереднє налаштування політик контролю.

16.2. Налаштування політики контролю мережевих каналів

Ефективність DLP-системи визначається не фактом її встановлення, а якістю налаштованих політик контролю. Політика контролю мережевого каналу типово визначає: перелік контрольованих каналів передавання даних (електронна пошта, веб-завантаження, месенджери, знімні носії, друк); правила виявлення чутливого вмісту в цих каналах (за ключовими словами, регулярними виразами, цифровими відбитками еталонних документів або категоріями вбудованого класифікатора); дію системи в разі виявлення порушення (лише журналювання й сповіщення офіцера безпеки або активне блокування передавання).

Практичний баланс між надмірно широкою політикою (яка генерує потік хибнопозитивних спрацювань, аналогічний уже розглянутій у розділі 12 проблемі систем виявлення вторгнень) і надмірно вузькою (яка пропускає реальні витоки) є основним щоденним завданням адміністратора DLP-системи й вимагає, як і адміністрування IDS/IPS, ітеративного тюнінгу правил за результатами реальної експлуатації, а не одноразового налаштування під час впровадження.

16.3. Розслідування інцидентів за допомогою повнотекстового індексу

Практична цінність архітектури SearchInform, описаної в підрозділі 16.1, розкривається повною мірою не в момент автоматичного спрацювання політики контролю, а під час розслідування вже виявленого чи запідозреного інциденту. На відміну від типової DLP-політики, яка лише блокує чи журналює конкретну подію передавання даних у момент її виникнення, централізований повнотекстовий індекс дозволяє офіцеру безпеки виконати **ретроспективний пошук**: знайти всі

повідомлення й документи певного співробітника за довільний період часу, що передував виявленню інциденту, встановити коло осіб, причетних до листування з певної теми, чи простежити, чи передавався конкретний конфіденційний документ раніше, до моменту, коли на нього спрацювала політика контролю.

Такий ретроспективний аналіз дозволяє відновити повну хронологію дій співробітника, аналогічно до того, як SIEM (розділ 3 модуля 1) відновлює хронологію дій зовнішнього зловмисника за журналами подій, і є тому природним доповненням процесу реагування на інциденти, розглянутого в підрозділі 10.5: якщо служба безпеки вже встановила факт витоку, повнотекстовий індекс SearchInform дає змогу оцінити реальний масштаб завданої шкоди, а не лише факт одного конкретного порушення.

16.4. Інтеграція SearchInform з іншими засобами моніторингу організації

DLP-система, розгорнута ізольовано від решти інфраструктури моніторингу, розглянутої в модулі 1, залишає офіцера безпеки без важливого контексту: подія DLP, що сама собою виглядає незначною (наприклад, одноразове копіювання невеликого файлу на знімний носій), істотно змінює пріоритет розслідування, якщо той самий користувач того ж дня став об'єктом сповіщення SIEM про підозрілу активність облікового запису. Тому в зрілій організації події DLP, включно з подіями SearchInform, експортуються до централізованої SIEM-платформи (Sentinel чи Splunk, розділи 4-5 модуля 1) як ще одне джерело подій поряд із мережевими журналами й журналами операційних систем, за тим самим конвеєром нормалізації, що розглядався в розділі 2.

Практично інтеграція реалізується або передаванням журналів SearchInform у SIEM через стандартний конектор чи syslog-переадресацію, або, рідше, зворотним напрямом, коли SIEM, виявивши підозрілу поведінку користувача засобами UEBA (підрозділ 3.7), автоматично підвищує рівень контролю DLP саме для цього користувача. Такий двонаправлений зв'язок перетворює DLP з ізольованого інструмента контролю каналів на повноцінного учасника загального контуру моніторингу організації, розглянутого в модулях 1-2 цієї книги. Питання того, як результати роботи всіх засобів моніторингу, включно з DLP, підлягають подальшому незалежному аудиту, розглядається в модулі 3.

16.5. Продуктивність і масштабування SearchInform у великих організаціях

Архітектура «агент-сервер», описана в підрозділі 16.1, добре справляється з кількома сотнями робочих станцій на єдиному сервері, але зі зростанням організації повнотекстовий індекс, що є основою ретроспективного пошуку з підрозділу 16.3, зростає пропорційно обсягу перехопленого трафіку й дедалі повільніше обробляє як нові надходження, так і самі пошукові запити офіцера безпеки. Практичне адміністрування великої інсталяції SearchInform передбачає ті самі рішення, що вже розглядалися для інших систем цього модуля: **розподілення навантаження** між кількома серверами індексації (концептуально паралельне до indexer clustering у Splunk, підрозділ 5.8 модуля 1), і **багаторівневе зберігання** (підрозділ 2.8 модуля 1) перехопленого контенту, коли свіжі дані лишаються в оперативному, швидкому індексі, а старіші, потрібні переважно для рідкісних ретроспективних розслідувань, переносяться в дешевше архівне сховище за визначеним політикою терміном (підрозділ 7.5 модуля 1).

Окремим викликом масштабування, специфічним саме для DLP, є зростання навантаження на самі кінцеві станції: агент, що перехоплює й локально обробляє значний обсяг активності користувача, за неправильного налаштування здатний помітно впливати на продуктивність робочої станції, що на практиці нерідко стає причиною скарг користувачів і тиску на службу безпеки з вимогою послабити політику контролю, тема, до якої повертається підрозділ 16.6.

16.6. Типові причини невдалого впровадження DLP

Значна частка проєктів впровадження DLP-систем, включно з SearchInform, не досягає заявленої мети захисту від витоку даних, і причини цього здебільшого не технічні. Найпоширенішою організаційною помилкою є впровадження DLP «мовчки», без попереднього роз'яснення співробітникам мети й меж контролю (підрозділ 7.5 модуля 1): раптове виявлення факту стеження підриває довіру в колективі сильніше, ніж сам факт контролю, і провокує пошук способів обходу політики (наприклад, через особисту, а не корпоративну пошту), що в результаті знижує реальну ефективність системи попри формально коректне технічне налаштування.

Другою поширеною причиною є вже згадана в підрозділі 16.5 надмірна вимогливість агента до ресурсів робочої станції, яка провокує обґрунтований опір користувачів і ІТ-підрозділу. Третьою, суто технічною причиною є політика контролю, що з самого початку налаштована або занадто широко (шквал

хибнопозитивних спрацювань, підрозділ 7.7 модуля 1, що виснажує ресурс аналітика вже в перші тижні експлуатації), або занадто вузько (система формально працює, але не виявляє реальних витоків, створюючи хибне відчуття захищеності). Успішні впровадження, на відміну від невдалих, як правило, починаються з обмеженого пілотного контуру (кілька підрозділів чи один канал контролю), поступового розширення охоплення й, що найважливіше, завчасної комунікації із зацікавленими сторонами ще до першого дня активного контролю.

16.7. Огляд альтернативних рішень на ринку DLP

SearchInform, розглянутий у цьому розділі як практична платформа, прямо згадана в робочій програмі дисципліни, не є єдиним рішенням класу DLP на ринку, і повний огляд теми був би неповним без короткого порівняльного контексту, за аналогією до огляду ринку SIEM-рішень у підрозділі 3.4 модуля 1. Серед інших постачальників, активних на пострадянському й східноєвропейському ринку, варто назвати **InfoWatch** і **Zecurion**, що пропонують подібну за функціональністю архітектуру «агент-сервер» із повнотекстовим пошуком. На глобальному ринку провідні позиції посідають **Forcepoint DLP**, **Symantec (Broadcom) DLP** і вбудовані можливості DLP великих екосистем, зокрема **Microsoft Purview DLP**, тісно інтегрований із хмарною екосистемою Microsoft 365, згаданою в контексті класифікації даних у підрозділі 7.8 модуля 1.

Вибір конкретного рішення на практиці визначається тими самими критеріями, що й вибір SIEM (таблиця 3.2 модуля 1): наявною інфраструктурою організації (глибока інтеграція з Microsoft 365 переважає вибір на користь Purview DLP для організацій, що вже живуть в цій екосистемі, аналогічно до логіки вибору Sentinel з підрозділу 4.1), сукупною вартістю володіння, і, для україномовного сегмента ринку, наявністю локалізованої підтримки продукту, якістю розпізнавання кириличного тексту (актуально для сигнатурного аналізу з таблиці 7.1 модуля 1) і відповідністю продукту вимогам, що можуть висуватися до постачальників програмного забезпечення для об'єктів критичної інфраструктури.

16.8. Юридичний супровід впровадження DLP: узгодження з профспілкою/трудовим колективом

Технічне розгортання DLP-системи, розглянуте в практикумі цього розділу, є лише частиною впровадження: DLP за своєю природою передбачає моніторинг дій працівника на робочому місці (аналіз документів, листування, файлових операцій),

що прямо зачіпає трудове законодавство й вимагає належного юридичного оформлення до, а не після технічного запуску системи. Впровадження засобів контролю без завчасного узгодження суперечить не лише етичним нормам, а й формальним вимогам трудового законодавства щодо повідомлення працівників про умови праці, які змінюються, та про запровадження засобів контролю за виконанням трудових обов'язків.

Практично це означає, що впровадженню DLP має передувати внесення відповідних положень до колективного договору чи правил внутрішнього трудового розпорядку (за наявності профспілкового органу, тобто за погодженням із профспілковим комітетом чи іншим уповноваженим представницьким органом трудового колективу), а також персональне ознайомлення кожного працівника з фактом і межами моніторингу під підпис до початку обробки його даних DLP-системою. Ця вимога прямо перетинається з принципами захисту персональних даних, розглянутими в джерелах цього розділу: працівник має право знати, які саме дані про нього збираються, з якою метою і хто має до них доступ, а обсяг моніторингу має відповідати принципу мінімізації й не виходити за межі, необхідні для законної мети захисту конфіденційної інформації організації.

Практикум 16. Розгортання політики контролю витоку конфіденційних документів

Мета роботи: набути практичних навичок налаштування політики контролю мережевих каналів у DLP-системі на прикладі захисту умовного набору конфіденційних документів організації.

Хід роботи:

1. у тестовому середовищі SearchInform (або аналогічного за архітектурою рішення) визначити набір з трьох-п'яти умовно конфіденційних документів і зняти цифрові відбитки їхнього вмісту;
2. налаштувати політику контролю, що виявляє спроби передавання вмісту цих документів (або істотної його частини) через контрольовані мережеві канали (електронна пошта, веб);
3. налаштувати додаткову політику на основі ключових слів для категорії даних, заданої викладачем (наприклад, персональні дані чи фінансова інформація);
4. відтворити в тестовому середовищі спробу передавання контрольованого вмісту й підтвердити коректне спрацювання обох політик за журналом подій;

5. свідомо підібрати граничний випадок (наприклад, передавання невеликого фрагмента документа або документа з незначними змінами) і оцінити, чи виявляє його налаштована політика, скоригувавши чутливість за потреби.

Контрольні питання до розділу 16

1. Яку архітектуру «агент-сервер» використовує SearchInform і за що відповідає кожен з цих компонентів?
2. Чим ретроспективний повнотекстовий пошук у зібраних даних відрізняється від сповіщення в реальному часі й для яких задач офіцера безпеки він потрібен?
3. Перелічіть типові канали передавання даних, які має контролювати політика DLP-системи.
4. Чому одноразового налаштування політики контролю під час впровадження, як правило, недостатньо для довгострокової ефективної роботи DLP-системи?
5. Порівняйте задачу тюнінгу хибнопозитивних спрацювань у DLP-системі із аналогічною задачею для IDS/IPS з розділу 12: що в них спільного?
6. Опишіть роль агента-перехоплювача на робочій станції в архітектурі SearchInform (рисунок 16.1).
7. Яку функцію виконує централізований повнотекстовий індекс сервера SearchInform?
8. Що визначає політика контролю мережевого каналу? Перелічіть щонайменше три складові.
9. Чим виявлення чутливого вмісту за цифровими відбитками відрізняється від виявлення за ключовими словами?
10. Чим відрізняється дія системи «лише журналювання» від дії «активне блокування»?
11. Чому надто широка політика контролю створює потік хибнопозитивних спрацювань, аналогічний проблемі IDS/IPS?
12. Які завдання адміністрування покладаються на офіцера інформаційної безпеки, що працює з SearchInform?
13. Наведіть приклад граничного випадку, який ускладнює налаштування політики контролю (наприклад, передавання невеликого фрагмента документа).
14. Чому одноразового налаштування політики DLP під час впровадження зазвичай недостатньо для довгострокової ефективності?
15. Які канали передавання даних, окрім електронної пошти, повинна контролювати повноцінна політика DLP?

16. Чим ретроспективний аналіз перехопленого трафіку доповнює можливості SIEM-системи з розділу 3?

МОДУЛЬ 3. АУДИТ ЗАХИЩЕНИХ ІТ-СИСТЕМ І МЕРЕЖ

Відповідає Змістовому модулю 3 робочої програми: Тема 5 «Аудит захищених ІТ систем і мереж та системи безпеки».

РОЗДІЛ 17. ОСНОВНІ ПІДХОДИ ДО МОНІТОРИНГУ ТА АУДИТУ БЕЗПЕКИ ІНФОРМАЦІЇ



*Розділ 17 · Основні підходи до
моніторингу та аудиту безпеки*

Зміст розділу

- 17.1. Аудит інформаційної безпеки: визначення й місце в циклі забезпечення безпеки*
- 17.2. Види аудиту: внутрішній, зовнішній, сертифікаційний*
- 17.3. Модель трьох ліній захисту (Three Lines Model)*
- 17.4. Об'єкти аудиту в захищеній ІТС*
- 17.5. Принципи проведення аудиту за ISO 19011*
- 17.6. Ролі учасників аудиту*
- 17.7. Аудит, комплаєнс і тестування на проникнення: розмежування суміжних понять*
- 17.8. Програма аудиту організації: багаторічне планування*
- 17.9. Ризик-орієнтоване планування аудиту: пріоритизація об'єктів за рівнем ризику*

Модулі 1 і 2 цієї книги привчали читача дивитися на захищену ІТС очима того, хто спостерігає за нею в реальному часі (моніторинг) або хто змінює її стан (адміністрування). Модуль 3 завершує коло, розпочате рисунком 1.1: **аудит** дивиться

на ту саму систему ретроспективно й незалежно, ставлячи запитання не «що відбувається зараз», а «чи справді все влаштовано так, як стверджують політики, журнали й звіти моніторингу». Без цього третього погляду перші два ризикують перетворитися на самозасвідчення: система, що сама себе моніторить і сама себе адмініструє, не має незалежного механізму виявити власні системні прогалини.

17.1. Аудит інформаційної безпеки: визначення й місце в циклі забезпечення безпеки

Аудит інформаційної безпеки — систематичний, незалежний і документований процес отримання доказів аудиту та їх об’єктивного оцінювання з метою визначення ступеня виконання узгоджених критеріїв аудиту (визначення узгоджується з термінологією стандарту ISO 19011, розглянутого детальніше в розділі 18). Три слова цього визначення несуть основне навантаження. **Систематичний** означає, що аудит виконується за заздалегідь визначеною методикою, а не як довільний, інтуїтивний огляд. **Незалежний** означає, що особа, яка проводить аудит, не оцінює результати власної роботи: адміністратор, що сам перевіряє коректність власних налаштувань, за визначенням не проводить аудиту в строгому сенсі цього слова. **Документований** означає, що висновки аудиту спираються на конкретні, простежувані докази, а не на загальне враження перевіряючого.

Підрозділ 1.1 модуля 1 уже розвів моніторинг, адміністрування й аудит як три складові єдиного циклу забезпечення безпеки (рисунок 1.1): моніторинг постачає дані в реальному часі, адміністрування усуває виявлені недоліки, аудит перевіряє результат на конкретний момент часу й ставить моніторингу нові завдання. Розділи цього модуля деталізують саме останню, третю складову цього циклу, яка досі згадувалася лише побіжно.

17.2. Види аудиту: внутрішній, зовнішній, сертифікаційний

Внутрішній аудит проводиться співробітниками самої організації (або аутсорсинговою командою, найнятою організацією для цієї ролі), підпорядкованими, як правило, вищому керівництву чи комітету з аудиту, а не підрозділу, що перевіряється. Мета внутрішнього аудиту, регулярне, повторюване запевнення керівництва, що заходи контролю (controls) працюють, як задумано, з можливістю оперативно виявляти й усувати прогалини до того, як вони переростуть в інцидент чи стануть предметом зовнішньої перевірки.

Зовнішній аудит проводиться незалежною третьою стороною, не пов'язаною трудовими відносинами з організацією, що перевіряється: аудиторською фірмою, регулятором, партнером по контракту, що вимагає перевірки постачальника. Зовнішній аудит має вищий рівень формальної незалежності, ніж внутрішній, і саме тому частіше вимагається законодавством чи договірними зобов'язаннями. **Сертифікаційний аудит**, окремий різновид зовнішнього аудиту, проводиться акредитованим органом сертифікації з конкретною метою: підтвердити відповідність системи менеджменту організації вимогам певного стандарту (найчастіше ISO/IEC 27001, детально розглянутого в підрозділі 18.2) і видати відповідний сертифікат, чинний протягом визначеного терміну за умови проходження щорічних наглядових аудитів.

17.3. Модель трьох ліній захисту (Three Lines Model)

Питання «хто саме повинен перевіряти захищеність ІТС» має усталену, широко визнану відповідь у формі **моделі трьох ліній (Three Lines Model)**, розробленої Інститутом внутрішніх аудиторів (ІІА) і зображеної на рисунку 17.1.



*Рис. 17.1. Модель трьох ліній захисту (Three Lines Model)
Джерело: розроблено автором на основі моделі Інституту внутрішніх
аудиторів (ІІА)*

Перша лінія — це операційний менеджмент і персонал, безпосередні власники ризику: адміністратор, що застосовує патч (підрозділ 9.8 модуля 2), аналітик SOC, що обробляє інцидент. Саме перша лінія щодня застосовує заходи контролю на практиці. **Друга лінія** об'єднує функції ризик-менеджменту й комплаєнсу: розробляє методологію, стандарти (підрозділ 9.2 модуля 2) і здійснює власний, не обов'язково незалежний моніторинг дотримання цих стандартів першою лінією. **Третя лінія**, внутрішній аудит, надає незалежне й об'єктивне запевнення керівництву й наглядовій раді щодо ефективності управління ризиками й заходів контролю обох попередніх ліній, не втручаючись безпосередньо в операційну діяльність. Зовнішній аудит і регулятор, показані окремо на рисунку 17.1, доповнюють цю модель ззовні, надаючи запевнення вже не лише керівництву самої організації, а й зовнішнім зацікавленим сторонам.

Практичний висновок моделі для читача цієї книги: більшість матеріалу модулів 1-2 (SOC-аналітик, адміністратор безпеки) стосується першої лінії; частина матеріалу про політики й стандарти (розділ 9 модуля 2) стосується другої; а весь модуль 3 присвячений компетенціям третьої лінії, аудиту.

17.4. Об'єкти аудиту в захищеній ІТС

Аудит захищеної ІТС рідко охоплює абсолютно все одночасно: кожна перевірка визначає власну **область аудиту (audit scope)**, конкретний перелік об'єктів, що підлягають оцінюванню. Типовими об'єктами є: технічна конфігурація систем (наприклад, відповідність налаштувань політикам паролів, розглянутим у практикумі до розділу 10 модуля 2); процеси й процедури (чи справді управління змінами, розділ 9 модуля 2, виконується так, як описано в документації, а не лише на папері); організаційні заходи (розподіл ролей і обов'язків, дотримання принципу найменших привілеїв, підрозділ 9.4 модуля 2); і, окремо, самі журнали та дані моніторингу (чи справді збираються ті події, які повинні збиратися, згідно з політикою покриття джерел, підрозділ 1.4 модуля 1).

Вибір об'єктів аудиту для конкретної перевірки визначається **критеріями аудиту (audit criteria)**: набором політик, процедур і вимог, з якими зіставляються зібрані докази. Без заздалегідь чітко визначених критеріїв аудит перетворюється на суб'єктивну оцінку аудитора, а не на об'єктивне зіставлення факту з вимогою.

17.5. Принципи проведення аудиту за ISO 19011

Стандарт ISO 19011 (докладніше в розділі 18) визначає сім принципів, яких повинен дотримуватися будь-який аудитор незалежно від предметної області аудиту: **добросовісність**, основа професіоналізму, виконання роботи чесно, сумлінно й відповідально; **об’єктивне подання**, обов’язок правдиво й точно відображати результати аудиту, включно з невирішеними розбіжностями думок; **належна професійна ретельність**, застосування старанності й розсудливості, пропорційної важливості завдання; **конфіденційність**, безпека отриманої під час аудиту інформації; **незалежність**, основа неупередженості й об’єктивності висновків, вже згадана в підрозділі 17.1; **підхід, заснований на доказах**, раціональний метод формулювання достовірних і відтворюваних висновків у систематичному процесі аудиту; і **ризик-орієнтований підхід**, що враховує ризики й можливості під час планування, проведення й звітування за результатами аудиту.

Останній принцип заслуговує на окреме пояснення, оскільки він визначає сучасний підхід до аудиту загалом: замість рівномірної, «килимової» перевірки абсолютно всього з однаковою інтенсивністю, ризик-орієнтований аудит свідомо спрямовує більше уваги й ресурсів на об’єкти з вищим потенційним впливом у разі невідповідності, залишаючи менш критичні об’єкти для менш глибокої, вибіркової перевірки.

17.6. Ролі учасників аудиту

Формальна термінологія аудиту розрізняє кількох дійових осіб. **Замовник аудиту (audit client)** — організація чи особа, що замовляє аудит, найчастіше вище керівництво чи комітет з аудиту. **Керівник групи аудиту (audit team leader)** відповідає за аудит загалом: планування, розподіл завдань між аудиторами, узгодження висновків. **Аудитор (auditor)** безпосередньо виконує перевірку конкретних об’єктів у межах своєї компетенції. **Аудований суб’єкт (auditee)** — організація чи підрозділ, діяльність якого перевіряється, і саме на аудованого суб’єкта покладається обов’язок надання доступу до необхідних доказів і персоналу.

Окремою, важливою для збереження незалежності роллю є **технічний експерт (technical expert)**: фахівець з вузькоспеціалізованих технічних питань (наприклад, конкретної хмарної платформи чи бази даних), що долучається до групи аудиту для консультації, але формально не є аудитором і не має повноважень формулювати висновки аудиту самостійно.

17.7. Аудит, комплаєнс і тестування на проникнення: розмежування суміжних понять

Три поняття, що часто плутають на практиці, заслуговують на чітке розмежування. **Комплаєнс (compliance)**, вже згаданий у підрозділі 3.9 модуля 1 стосовно звітності SIEM, є постійною, здебільшого автоматизованою функцією підтримання відповідності вимогам у реальному часі, тоді як **аудит** є точковою, періодичною перевіркою цієї відповідності незалежною стороною: комплаєнс каже «ми відповідаємо вимогам», аудит перевіряє, чи це твердження справді правдиве. **Тестування на проникнення (penetration testing)**, коротко згадане в підрозділі 12.9 модуля 2 у контексті адверсарної симуляції, технічно й методологічно найближче до аудиту з погляду незалежності перевіряючого, але переслідує іншу мету: не оцінити відповідність процесів і документації встановленим критеріям, а практично довести (чи спростувати) можливість експлуатації конкретних технічних вразливостей, часто в межах узгодженого, обмеженого технічного обсягу, а не всієї системи керування безпекою організації.

Практичний наслідок цього розмежування: зріла організація не обирає між аудитом, комплаєнсом і тестуванням на проникнення, а поєднує всі три, оскільки кожен з них покриває свою, лише частково перекривну частину загальної картини захищеності.

17.8. Програма аудиту організації: багаторічне планування

Окрема аудиторська перевірка, детально розглянута в розділі 19, є лише одним елементом ширшої **програми аудиту (audit programme)**: багаторічного плану, що визначає, які саме об'єкти й процеси, з якою періодичністю й у якій послідовності підлягають аудиту протягом визначеного горизонту планування, зазвичай одного-трьох років. Побудова програми аудиту сама спирається на ризик-орієнтований принцип (підрозділ 17.5): критичніші системи (наприклад, контролер домену, підрозділ 11.1 модуля 2, чи публічно доступні сервіси) аудитуються частіше, ніж другорядні внутрішні інструменти.

Формування програми аудиту є відповідальністю функції внутрішнього аудиту (третя лінія, підрозділ 17.3) і, як правило, затверджується комітетом з аудиту чи наглядовою радою організації. Обмеженість ресурсів функції аудиту (кількість кваліфікованих аудиторів, час) майже завжди означає, що програма не здатна охопити абсолютно всі можливі об'єкти щороку, тому саме якість пріоритизації, а

не формальна повнота охоплення, визначає практичну цінність програми аудиту для організації.

17.9. Ризик-орієнтоване планування аудиту: пріоритизація об'єктів за рівнем ризику

Підрозділ 17.8 констатував, що ресурси аудиту завжди обмежені, а отже потребують пріоритизації; розглянемо, як ця пріоритизація здійснюється практично. Ризик-орієнтоване планування (risk-based audit planning) будує чергу об'єктів аудиту не за алфавітом чи зручністю, а на основі оцінки ризику кожного об'єкта, зазвичай як добутку ймовірності реалізації загрози й потенційного впливу на організацію у разі її реалізації, за логікою, спорідненою з оцінюванням ризиків у процесі управління вразливостями (підрозділ 20.2). Джерелами вхідних даних для такої оцінки слугують результати попередніх аудитів (розділ 19), реєстр інцидентів (розділ 3 модуля 1) і критичність активу для безперервності бізнесу, визначена власником процесу.

Практичний результат ризик-орієнтованого підходу, зазвичай оформлений як **теплова карта ризиків (risk heat map)**, дозволяє наочно розподілити об'єкти аудиту за квадрантами «високий вплив/висока ймовірність» і подібними, спрямовуючи обмежений ресурс аудиторської команди насамперед на системи, розташовані у квадранті найвищого пріоритету, наприклад, системи, що обробляють персональні дані клієнтів (підрозділ 7.8 модуля 1) чи керують критичною інфраструктурою. Такий підхід не заперечує потреби періодично, хоча й рідше, перевіряти й системи з нижчим оціненим ризиком, оскільки сама оцінка ризику з часом змінюється і потребує регулярного перегляду.

Практикум 17. Розробка фрагмента програми аудиту інформаційної безпеки

Мета роботи: застосувати ризик-орієнтований підхід (підрозділ 17.5) для формування пріоритизованого фрагмента річної програми аудиту умовної організації.

Хід роботи:

1. визначити перелік із щонайменше восьми об'єктів аудиту умовної організації, спираючись на матеріал модулів 1-2 (наприклад: контролер домену, конфігурація міжмережевого екрана, політика паролів, налаштування DLP, хмарне сховище даних, кластер Kubernetes);
2. для кожного об'єкта оцінити орієнтовний рівень ризику (високий/середній/низький) за критичністю активу (підрозділ 9.7 модуля 2) і ймовірною частотою змін конфігурації;
3. розподілити об'єкти за трьома пріоритетними групами відповідно до оцінки ризику;
4. запропонувати періодичність аудиту для кожної групи (наприклад, щоквартально для високого ризику, щорічно для низького);
5. оформити результат як фрагмент програми аудиту у вигляді таблиці «об'єкт — рівень ризику — періодичність — відповідальний».

Контрольні питання до розділу 17

1. Дайте означення аудиту інформаційної безпеки і поясніть значення слів «систематичний», «незалежний», «документований» у цьому визначенні.
2. Чим внутрішній аудит принципово відрізняється від зовнішнього?
3. Що таке сертифікаційний аудит і чим він відрізняється від звичайного зовнішнього аудиту?
4. Опишіть модель трьох ліній захисту (рисунок 17.1) і роль кожної лінії.
5. До якої лінії моделі належить SOC-аналітик? А внутрішній аудитор?
6. Які типові об'єкти аудиту в захищеній ІТС?
7. Що таке критерії аудиту (audit criteria) і чому аудит неможливий без їх заздалегідь чіткого визначення?
8. Перелічіть сім принципів аудиту за ISO 19011.
9. У чому полягає ризик-орієнтований підхід до аудиту на відміну від рівномірної «килимової» перевірки?

10. Опишіть ролі замовника аудиту, керівника групи аудиту, аудитора й аудованого суб'єкта.
11. Яку роль в аудиті виконує технічний експерт і чому він не є аудитором?
12. Чим аудит відрізняється від комплаєнсу?
13. Чим аудит відрізняється від тестування на проникнення за метою й методом?
14. Чому зріла організація поєднує аудит, комплаєнс і тестування на проникнення, а не обирає щось одне?
15. Що таке програма аудиту організації і на який горизонт планування вона зазвичай розраховується?
16. Чому програма аудиту рідко здатна охопити абсолютно всі об'єкти організації щороку?
17. Наведіть приклад того, як принцип незалежності аудиту порушується, якщо адміністратор перевіряє власні налаштування.

РОЗДІЛ 18. СТАНДАРТИ ТА МЕТОДИКИ АУДИТУ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ



Зміст розділу

- 18.1. ISO 19011: настанови щодо аудиту систем менеджменту**
- 18.2. ISO/IEC 27001 і цикл сертифікаційного аудиту**
- 18.3. NIST SP 800-53A: оцінювання заходів безпеки**
- 18.4. Українська нормативна база аудиту інформаційної безпеки**
- 18.5. Галузеві стандарти аудиту: PCI DSS та SOC 2**
- 18.6. Чекліст-орієнтований проти ризик-орієнтованого аудиту**
- 18.7. Матриця відповідності між стандартами (compliance mapping)**
- 18.8. Вибір методики аудиту під конкретну задачу організації**
- 18.9. Аудит третіх сторін і постачальників (third-party/vendor audit)**

Розділ 17 визначив, що таке аудит і хто його проводить. Це запитання логічно тягне за собою наступне: за яким саме еталоном порівнювати те, що аудитор бачить у системі. Відповідь дає не один, а ціле сімейство взаємодоповнюючих стандартів і методик, кожен з яких має власну сферу застосування, історію походження й ступінь юридичної обов’язковості для української організації.

18.1. ISO 19011: настанови щодо аудиту систем менеджменту

ISO 19011 — міжнародний стандарт, що визначає настанови щодо аудиту систем менеджменту загалом, незалежно від предметної області (якість, екологія, інформаційна безпека). Саме на цей стандарт спиралося визначення аудиту й сім принципів аудиту, розглянуті в підрозділах 17.1 і 17.5. Важливо розуміти статус ISO 19011: це не стандарт, відповідність якому сертифікується (на відміну від ISO/IEC 27001, підрозділ 18.2), а методологічний документ, що описує, як правильно планувати, проводити й документувати аудит будь-якої системи менеджменту, включно з системою менеджменту інформаційної безпеки.

Практична цінність ISO 19011 для книги в цілому в тому, що саме цей стандарт формує спільну мову й методику для розділу 19 (етапи аудиторської перевірки безпосередньо повторюють життєвий цикл, визначений ISO 19011).

18.2. ISO/IEC 27001 і цикл сертифікаційного аудиту

ISO/IEC 27001, вже неодноразово згаданий у попередніх розділах (зокрема підрозділ 9.2 модуля 2) як джерело вимог до системи менеджменту інформаційної безпеки (СМІБ, ISMS), є, ймовірно, найпоширенішим стандартом, під який організації проходять сертифікаційний аудит (підрозділ 17.2). Цикл сертифікації складається з кількох послідовних етапів: **аудит першого етапу (Stage 1)**, попередній огляд документації СМІБ і готовності організації до основного аудиту; **аудит другого етапу (Stage 2)**, змістовна перевірка фактичного функціонування заходів контролю, зазначених у документації, з виїздом аудиторів на місце; за успішного проходження обох етапів видається **сертифікат**, чинний, як правило, три роки; протягом цього періоду щорічно проводяться **наглядові аудити (surveillance audits)**, вужчі за обсягом, які підтверджують, що СМІБ і надалі функціонує належним чином; після завершення трирічного циклу проводиться **ресертифікаційний аудит**, за обсягом близький до початкового Stage 2.

Важливо розрізняти сам стандарт ISO/IEC 27001 (вимоги до системи менеджменту) і супровідний стандарт **ISO/IEC 27002** (детальний каталог заходів

контролю та настанов щодо їх впровадження, вже згаданий у підрозділі 9.1 модуля 2): 27001 визначає, що організація повинна мати (систему управління ризиками, процес постійного вдосконалення), тоді як 27002 радить, як саме реалізувати конкретні технічні й організаційні заходи контролю.

18.3. NIST SP 800-53A: оцінювання заходів безпеки

Якщо ISO/IEC 27001 переважає в міжнародній, зокрема європейській практиці, то в контексті федеральних інформаційних систем США й, дедалі частіше, у контексті постачальників, що працюють із такими системами, панує сімейство стандартів NIST. **NIST SP 800-53**, вже згаданий у підрозділі 9.9 модуля 2 стосовно управління вразливостями, визначає власне каталог заходів контролю безпеки й приватності. Пряме доповнення до нього, **NIST SP 800-53A (Assessment Procedures)**, детально описує саме методику оцінювання (аудиту) того, чи справді кожен конкретний захід контролю з каталогу 800-53 реалізовано ефективно: для кожного заходу контролю стандарт пропонує конкретні процедури оцінювання (перевірку документації, опитування персоналу, технічне тестування), концептуально паралельні до методів збору доказів, розглянутих у підрозділі 19.3.

Практична відмінність підходу NIST від ISO/IEC 27001 полягає в більшій деталізації й приписовості (prescriptiveness) конкретних технічних вимог, тоді як ISO/IEC 27001 залишає організації більше свободи у виборі конкретних заходів контролю за умови досягнення заявлених цілей управління ризиками.

18.4. Українська нормативна база аудиту інформаційної безпеки

Підрозділ 1.3 модуля 1 уже окреслив українську нормативно-правову базу моніторингу; для аудиту актуальні ті самі рівні регулювання з додатковою специфікою. На рівні національних стандартів **ДСТУ ISO/IEC 27001:2023**, гармонізована з міжнародним оригіналом версія, дозволяє українським організаціям проходити сертифікацію з урахуванням національної практики застосування. Окремим, історично сформованим шаром лишаються **нормативні документи системи технічного захисту інформації (НД ТЗІ)**, що визначають вимоги до захисту інформації в державних інформаційних системах і на об'єктах критичної інфраструктури, включно з власною методикою експертизи (державної експертизи) відповідності комплексної системи захисту інформації встановленим вимогам, окремим, історично сформованим інститутом аудиту, паралельним до міжнародної сертифікації ISO/IEC 27001.

Для об'єктів критичної інфраструктури обов'язковість проходження державної експертизи чи аудиту визначається профільним законодавством (зокрема Законом України «Про основні засади забезпечення кібербезпеки України», згаданим у підрозділі 1.3 модуля 1), тоді як для решти організацій сертифікація ISO/IEC 27001 чи проходження аудиту NIST лишається добровільним, договірно чи ринково мотивованим рішенням.

18.5. Галузеві стандарти аудиту: PCI DSS та SOC 2

Поряд із загальними стандартами системи менеджменту існують галузеві стандарти, обов'язкові лише для організацій певного профілю діяльності. **PCI DSS (Payment Card Industry Data Security Standard)** обов'язковий для будь-якої організації, що обробляє, зберігає чи передає дані платіжних карток, і визначає детальний, приписовий перелік технічних вимог (зокрема, обов'язковість File Integrity Monitoring, детально розглянутого в розділі 21, і сегментації мережі, підрозділ 10.4 модуля 2, для ізоляції середовища обробки карткових даних). Аудит відповідності PCI DSS проводить кваліфікований оцінювач безпеки (QSA, Qualified Security Assessor), і залежно від обсягу оброблюваних транзакцій організація проходить або повний аудит на місці, або скорочену процедуру самооцінювання.

SOC 2 (System and Organization Controls 2), звіт, що видається за результатами аудиту, проведеного незалежним аудитором за стандартами Американського інституту сертифікованих публічних бухгалтерів (AICPA), набув особливого поширення серед постачальників хмарних послуг і SaaS-застосунків (підрозділ 14.1 модуля 2) як спосіб надати клієнтам незалежне запевнення щодо безпеки, доступності й конфіденційності послуги без розкриття внутрішніх деталей інфраструктури постачальника. Звіт SOC 2 типу II, найпоширеніший на практиці, підтверджує ефективність заходів контролю не в одній конкретній точці часу, а протягом визначеного періоду спостереження (типово шести-дванадцяти місяців).

18.6. Чекліст-орієнтований проти ризик-орієнтованого аудиту

Практичне виконання аудиту за будь-яким із розглянутих стандартів може будуватися за одним із двох принципово різних підходів. **Чекліст-орієнтований аудит** послідовно перевіряє кожен пункт наперед складеного переліку вимог, незалежно від його відносної важливості для конкретної організації: підхід простий у виконанні й забезпечує формальну повноту покриття, але ризикує витратити

непропорційно багато часу на другорядні пункти, залишаючи менше ресурсу на глибоке дослідження найкритичніших ризиків.

Ризик-орієнтований аудит, логічне продовження принципу з підрозділу 17.5, спрямовує основні зусилля аудитора на об'єкти й процеси з найвищим потенційним впливом у разі невідповідності, свідомо застосовуючи менш глибоку перевірку до об'єктів низького ризику. Сучасна практика, включно з методологією самого ISO 19011, дедалі більше тяжіє до ризик-орієнтованого підходу, хоча чекліст (у формі детального переліку контрольних питань чи вимог стандарту) лишається практичним інструментом ризик-орієнтованого аудиту, а не його протилежністю: різниця радше в тому, як розподіляється час і глибина перевірки між пунктами чекліста, а не у факті наявності самого чекліста.

18.7. Матриця відповідності між стандартами (compliance mapping)

Організація, що прагне (чи зобов'язана) відповідати кільком стандартам одночасно, наприклад ISO/IEC 27001 для міжнародних клієнтів і НД ТЗІ для державного замовника, стикається з практичною проблемою: вимоги різних стандартів частково перекриваються, а частково унікальні для кожного. **Матриця відповідності (compliance mapping, crosswalk)** зіставляє вимоги різних стандартів у єдину таблицю, показуючи, який захід контролю одного стандарту покриває (повністю чи частково) відповідну вимогу іншого, а яка вимога унікальна й потребує окремого, додаткового заходу.

Практична цінність такого зіставлення подвійна: по-перше, воно дозволяє уникнути дублювання роботи, коли один і той самий технічний захід контролю (наприклад, шифрування журналів, підрозділ 2.6 модуля 1) підтверджується один раз, а результат використовується для звітності одразу за кількома стандартами; по-друге, воно наочно показує прогалини, вимоги, не покриті жодним із наявних заходів контролю, які потребують окремої уваги при плануванні програми аудиту (підрозділ 17.8).

18.8. Вибір методики аудиту під конкретну задачу організації

Розглянуті в цьому розділі стандарти й методики не конкурують за єдине «правильне» місце, а застосовуються обрано, залежно від конкретної мети перевірки. Організація, що готується до продажу послуг європейському клієнтові, найімовірніше обиратиме шлях сертифікації ISO/IEC 27001 (підрозділ 18.2). Постачальник хмарного SaaS-рішення для американського ринку частіше обиратиме

звіт SOC 2 (підрозділ 18.5). Об'єкт критичної інфраструктури України зобов'язаний виконувати вимоги НД ТЗІ (підрозділ 18.4) незалежно від власних комерційних переваг. Організація, що приймає платежі картками, зобов'язана пройти аудит PCI DSS (підрозділ 18.5) незалежно від решти обраних стандартів.

Практичне рішення про методику аудиту, таким чином, рідко буває суто технічним: воно визначається комбінацією юридичних зобов'язань, вимог ринку й контрагентів, і лише потім, у межах цих обмежень, власними уявленнями організації про найкращу практику управління ризиками.

18.9. Аудит третіх сторін і постачальників (third-party/vendor audit)

Стандарти й методики, розглянуті в цьому розділі, звично застосовуються до власної інфраструктури організації, проте сучасний ланцюг постачання рідко обмежується лише нею: хмарний провайдер (розділ 14 модуля 2), аутсорсингова служба підтримки чи постачальник програмного забезпечення для об'єкта критичної інфраструктури так само здатні стати джерелом інциденту безпеки, як і власні системи, що й ілюструє модель спільної відповідальності (підрозділ 14.3 модуля 2). **Аудит третіх сторін (third-party audit)** поширює принципи цього розділу на постачальників, з якими організація ділиться даними чи від чиєї інфраструктури залежить її власна безперервність.

На практиці аудит третіх сторін рідко означає повноцінну незалежну перевірку кожного постачальника власними силами: значно поширенішим є запит і аналіз наявного звіту незалежного аудитора (найчастіше SOC 2 Type II, підрозділ 18.5, чи сертифіката ISO/IEC 27001, підрозділ 18.2), доповнений опитувальником безпеки постачальника (vendor security questionnaire) і, для критичних постачальників, правом на аудит (right to audit), закріпленим безпосередньо в договорі. Періодичність і глибина перевірки постачальника, за аналогією з ризик-орієнтованим плануванням (підрозділ 17.9), мають визначатися критичністю даних чи сервісів, які цьому постачальнику довірено, а не однаковим для всіх формальним підходом.

Практикум 18. Побудова матриці відповідності між ISO/IEC 27001 і НД ТЗІ

Мета роботи: набути навичок зіставлення вимог різних стандартів аудиту інформаційної безпеки.

Хід роботи:

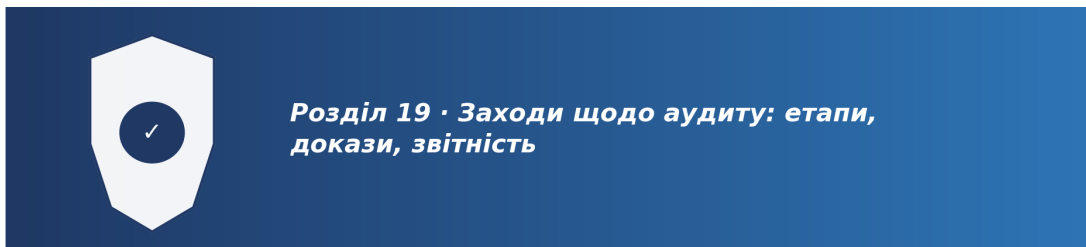
1. обрати п'ять контрольних вимог із додатка А стандарту ISO/IEC 27001 (наприклад, керування доступом, журналювання подій, управління вразливостями, резервне копіювання, реагування на інциденти);
2. для кожної вимоги знайти відповідний або найближчий за змістом пункт чинної нормативної бази технічного захисту інформації в Україні;
3. позначити ступінь відповідності для кожної пари (повна, часткова, відсутня);
4. для пар із частковою або відсутньою відповідністю сформулювати, який додатковий захід контролю знадобиться, щоб закрити прогалину;
5. оформити результат у вигляді таблиці «вимога ISO/IEC 27001 — відповідний пункт НД ТЗІ — ступінь відповідності — необхідні додаткові заходи».

Контрольні питання до розділу 18

1. Який статус має стандарт ISO 19011: чи сертифікується відповідність йому і чому?
2. Опишіть етапи сертифікаційного циклу ISO/IEC 27001: Stage 1, Stage 2, наглядові аудити, ресертифікація.
3. Чим стандарт ISO/IEC 27001 відрізняється від ISO/IEC 27002?
4. Що таке NIST SP 800-53A і як він пов'язаний з каталогом NIST SP 800-53?
5. Назвіть рівні української нормативної бази, релевантні для аудиту інформаційної безпеки.
6. Що таке НД ТЗІ і для яких організацій їх виконання обов'язкове?
7. Для яких організацій обов'язковий аудит PCI DSS?
8. Чим звіт SOC 2 типу II відрізняється від одноразової перевірки на конкретну дату?
9. Порівняйте чекліст-орієнтований і ризик-орієнтований підходи до аудиту.
10. Чи є чекліст протилежністю ризик-орієнтованого підходу? Обґрунтуйте відповідь.
11. Що таке матриця відповідності (compliance mapping) і яку практичну проблему вона вирішує?

12. Наведіть приклад заходу контролю, що одночасно покриває вимоги двох різних стандартів.
13. Якими факторами визначається вибір методики аудиту для конкретної організації?
14. Чому вибір стандарту аудиту рідко є суто технічним рішенням?
15. Порівняйте підхід NIST (каталог 800-53/800-53A) і підхід ISO/IEC 27001 за рівнем деталізації вимог.
16. Що таке QSA і яку роль ця роль відіграє в аудиті PCI DSS?

РОЗДІЛ 19. ЗАХОДИ ЩОДО АУДИТУ ЗАХИЩЕНИХ ІТ-СИСТЕМ: ЕТАПИ, ЗБІР ДОКАЗІВ, ЗВІТНІСТЬ



Зміст розділу

- 19.1. Життєвий цикл аудиторської перевірки***
- 19.2. Планування аудиту й визначення області охоплення***
- 19.3. Методи збору доказів аудиту***
- 19.4. Робочі документи аудитора***
- 19.5. Класифікація невідповідностей за критичністю***
- 19.6. Складання звіту аудиту***
- 19.7. План коригувальних дій і подальший контроль***
- 19.8. Забезпечення якості аудиту***
- 19.9. Комунікація результатів аудиту з керівництвом (closing meeting)***

Розділи 17-18 відповіли на запитання «що таке аудит» і «за яким стандартом його проводити». Цей розділ переходить до практичної механіки: як саме окрема аудиторська перевірка розгортається в часі, від першого контакту із замовником до

завершального звіту, і які конкретні артефакти, докази, робочі документи, звіт, вона повинна залишити по собі.

19.1. Життєвий цикл аудиторської перевірки

ISO 19011 (підрозділ 18.1) визначає життєвий цикл окремої аудиторської перевірки як послідовність п'яти етапів, зображених на рисунку 19.1.



Рис. 19.1. Життєвий цикл аудиторської перевірки за ISO 19011
Джерело: розроблено автором на основі методики ISO 19011

Ініціювання аудиту визначає мету, обсяг і критерії конкретної перевірки та встановлює перший контакт з аудованим суб'єктом. **Підготовка діяльності** охоплює формування групи аудиту, розподіл ролей і складання плану аудиту

(детальніше підрозділ 19.2). **Проведення аудиту** — власне збір доказів на місці чи дистанційно (підрозділ 19.3). **Підготовка та розсилання звіту** документує зібрані докази й сформульовані висновки (підрозділ 19.6). **Завершення аудиту** включає архівування робочих документів і, за потреби, подальший контроль виконання коригувальних дій (підрозділ 19.7). Пунктирне коло на рисунку 19.1 підкреслює, що окрема перевірка не є ізольованою подією, а вбудована в безперервну програму аудиту, вже розглянуту в підрозділі 17.8.

19.2. Планування аудиту й визначення області охоплення

Етап підготовки з підрозділу 19.1 завершується формальним **планом аудиту (audit plan)**: документом, що фіксує узгоджені з підрозділу 17.4 критерії й область аудиту, склад групи аудиторів (з урахуванням компетенції й незалежності від об'єкта перевірки), орієнтовний графік проведення, і, за потреби, логістичні деталі (доступ до приміщень, систем, потрібного персоналу аудитованого суб'єкта).

Планування також визначає **вибірку (sampling)**: коли повна перевірка всіх без винятку об'єктів у межах області аудиту фізично неможлива (наприклад, перевірка конфігурації кожного з тисяч робочих станцій), аудитор обирає репрезентативну вибірку, розмір і метод формування якої (випадкова, стратифікована за критичністю активу з підрозділу 9.7 модуля 2) повинні бути обґрунтовані й задокументовані, оскільки саме на цьому етапі найлегше, свідомо чи несвідомо, спотворити результат аудиту вибором зручної, а не репрезентативної вибірки.

19.3. Методи збору доказів аудиту

Доказ аудиту (audit evidence) — записи, викладення фактів або інша інформація, що є доречною для критеріїв аудиту і може бути перевірена. Основні методи збору доказів: **інтерв'ю** з персоналом аудитованого суб'єкта, з'ясування того, як процес виконується на практиці, а не лише як він описаний у документації; **спостереження** за фактичним виконанням процесу (наприклад, спостереження за тим, як адміністратор насправді виконує процедуру створення облікового запису); **документальна перевірка**, аналіз політик, журналів, протоколів, конфігураційних файлів; і **технічне тестування**, безпосередня технічна перевірка налаштувань системи (наприклад, підтвердження, що політика паролів, підрозділ 10.4 модуля 2, справді технічно застосована, а не лише задекларована в документі).

Ключовий принцип, що об'єднує всі чотири методи: доказ, отриманий лише одним методом, зазвичай має нижчу переконливість, ніж доказ, підтверджений

кількома незалежними джерелами (**тріангуляція доказів**). Наприклад, твердження адміністратора під час інтерв'ю про те, що патч-менеджмент (підрозділ 9.8 модуля 2) виконується регулярно, набагато переконливіше, якщо воно підтверджується документальним переглядом реєстру змін і технічною перевіркою фактичного рівня оновлення вибірки систем.

19.4. Робочі документи аудитора

Кожен доказ, зібраний за методами підрозділу 19.3, повинен бути зафіксований у **робочих документах аудитора (working papers)**: структурованому наборі записів, скріншотів, копій документів, протоколів інтерв'ю, що обґрунтовують кожен висновок звіту аудиту. Робочі документи виконують ту саму функцію для аудитора, що ланцюг зберігання доказів (підрозділ 2.10 модуля 1) виконує для розслідування інциденту: вони дозволяють будь-якій третій стороні (наприклад, рецензенту з підрозділу 19.8 чи наступному аудитору через рік) простежити, на підставі яких саме фактів було зроблено конкретний висновок, не покладаючись лише на слово аудитора.

Практична вимога до якості робочих документів: кожен запис повинен однозначно вказувати, хто, коли й яким методом зібрав конкретний доказ, а сам доказ повинен бути достатньо деталізованим, щоб інша кваліфікована особа, не присутня під час первинного збору, могла дійти того самого висновку на його підставі.

19.5. Класифікація невідповідностей за критичністю

Не кожне відхилення від критеріїв аудиту (підрозділ 17.4) однаково небезпечне, тому виявлені **невідповідності (findings, nonconformities)** класифікують за критичністю. Типова шкала включає: **критичну невідповідність**, що вказує на системний збій заходу контролю з високою ймовірністю прямого й суттєвого впливу на захищеність (наприклад, повна відсутність журналювання критичної системи); **значну невідповідність**, порушення вимоги стандарту без негайного критичного впливу, але з реальним ризиком, якщо не буде усунуто; **незначну невідповідність**, дрібне, локальне відхилення, що не ставить під загрозу загальну ефективність системи контролю; і **спостереження (observation)**, зауваження, що формально не є порушенням вимоги, але вказує на потенційну можливість для вдосконалення.

Класифікація безпосередньо впливає на терміновість і формат подальшої реакції (підрозділ 19.7): критична невідповідність, як правило, вимагає негайного інформування керівництва ще до завершення повного звіту, тоді як спостереження може бути просто зазначене в переліку рекомендацій без окремого плану коригувальних дій.

19.6. Складання звіту аудиту

Звіт аудиту є основним, формальним результатом усієї перевірки й типово містить: мету, область і критерії аудиту, узгоджені на етапі планування (підрозділ 19.2); методологію та період проведення; перелік виявлених невідповідностей з класифікацією за критичністю (підрозділ 19.5), з посиланням на конкретні докази з робочих документів (підрозділ 19.4); загальний висновок аудитора щодо рівня відповідності об'єкта перевірки встановленим критеріям; і рекомендації щодо усунення виявлених невідповідностей.

Принцип **об'єктивного подання**, вже згаданий серед принципів аудиту в підрозділі 17.5, вимагає, щоб звіт правдиво відображав результати, включно з позитивними висновками (заходи контролю, що працюють належним чином), а не лише переліком проблем: звіт, що складається виключно з переліку недоліків, ризикує сприйматися аудованим суб'єктом як несправедливий і підриває конструктивність подальшої взаємодії довкола коригувальних дій.

19.7. План коригувальних дій і подальший контроль

Виявлення невідповідності саме по собі не усуває проблему: за нею повинен слідувати **план коригувальних дій (Corrective Action Plan, CAP)**, документ, розроблений аудованим суб'єктом (а не аудитором, чия роль обмежується виявленням і рекомендаціями, а не безпосереднім виправленням, з міркувань збереження незалежності, підрозділ 17.5), що визначає конкретні кроки усунення кожної невідповідності, відповідальну особу і термін виконання.

Функція аудиту (найчастіше друга чи третя лінія моделі з підрозділу 17.3) здійснює **подальший контроль (follow-up)**: перевіряє, чи справді заплановані коригувальні дії виконано в узгоджений термін, зазвичай через окрему, вужчу за обсягом перевірку, а не покладаючись лише на самозвіт аудованого суб'єкта. Невиконаний план коригувальних дій, що переноситься з циклу в цикл без реального прогресу, є класичною ознакою слабкої культури управління ризиками в організації і, як правило, ескалюється до вищого рівня керівництва.

19.8. Забезпечення якості аудиту

Аудит, що сам не підлягає жодній перевірці якості, ризикує поступово втратити методологічну строгість. **Забезпечення якості (quality assurance)** функції аудиту охоплює **внутрішнє рецензування (внутрішній peer review)**, коли робочі документи й звіт одного аудитора перевіряються іншим, досвідченішим членом команди перед фінальним затвердженням, і періодичну **зовнішню оцінку** самої функції внутрішнього аудиту незалежною стороною, що підтверджує відповідність її роботи професійним стандартам (наприклад, Міжнародним професійним практикам ПА).

Практичний зв'язок із рештою розділу: без рецензування помилки на будь-якому з попередніх етапів, неправильно визначена область аудиту (підрозділ 19.2), недостатньо переконливі докази (підрозділ 19.3), необґрунтована класифікація критичності (підрозділ 19.5), можуть залишитися непоміченими аж до моменту, коли вони підірвуть довіру до всього звіту аудиту в очах керівництва чи зовнішньої сторони.

19.9. Комунікація результатів аудиту з керівництвом (closing meeting)

Навіть технічно бездоганний звіт аудиту, розглянутий у підрозділі 19.6, не досягає мети, якщо його висновки не донесені до керівництва в спосіб, що спонукає до реальних коригувальних дій. **Закриваюча нарада (closing meeting)**, що проводиться до фіналізації звіту, дає аудиторам можливість представити попередні висновки власникам перевіреного процесу, вислухати їхні коментарі й уточнення щодо контексту знахідок, і в разі потреби скоригувати класифікацію критичності (підрозділ 19.5) до того, як звіт набуде офіційного статусу, зменшуючи ризик формальних заперечень на пізнішому етапі.

Ефективна комунікація результатів аудиту з керівництвом вищого рівня, на відміну від технічної комунікації з власником процесу, вимагає перекладу технічних знахідок мовою бізнес-ризиків: замість формулювання «виявлено застарілу версію бібліотеки в компоненті X» ефективніший звіт для керівництва формулює наслідок «наявна вразливість, що за певних умов дозволяє зловмиснику отримати доступ до персональних даних клієнтів, з відповідним репутаційним і регуляторним ризиком». Саме така мова, поєднана з чіткими рекомендаціями й узгодженими термінами усунення (підрозділ 19.7), перетворює звіт аудиту з формального документа на інструмент реального управління ризиками організації.

Практикум 19. Проведення навчального аудиту фрагмента конфігурації безпеки

Мета роботи: застосувати повний життєвий цикл аудиторської перевірки (рисунок 19.1) до обмеженого, навчального об'єкта.

Хід роботи:

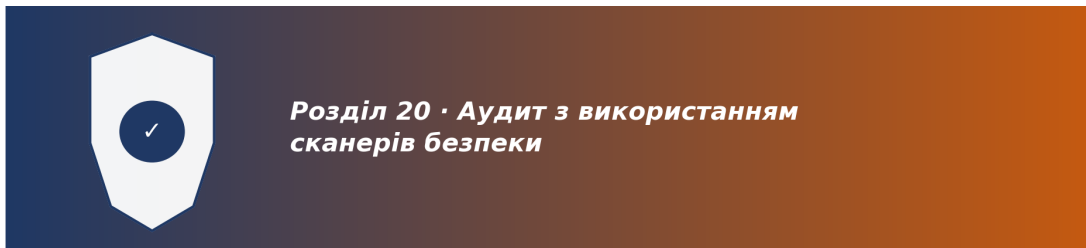
1. визначити область аудиту (наприклад, політика паролів і мережева сегментація тестового середовища, за матеріалами практикуму до розділу 10 модуля 2) і критерії аудиту (конкретний перелік вимог, з якими звірятиметься фактична конфігурація);
2. скласти короткий план аудиту: склад умовної групи, орієнтовний графік, метод вибірки;
3. зібрати докази щонайменше двома різними методами з підрозділу 19.3 (наприклад, документальна перевірка політики та технічна перевірка фактичного налаштування);
4. оформити знайдені відхилення як невідповідності з класифікацією за критичністю (підрозділ 19.5), кожна з яких супроводжується конкретним доказом;
5. скласти короткий звіт аудиту за структурою підрозділу 19.6 і додати до нього орієнтовний план коригувальних дій.

Контрольні питання до розділу 19

1. Перелічіть п'ять етапів життєвого циклу аудиторської перевірки за ISO 19011.
2. Що визначає план аудиту і хто його затверджує?
3. Що таке вибірка (sampling) в аудиті і чому вона часто необхідна?
4. Перелічіть чотири основні методи збору доказів аудиту.
5. Що таке тріангуляція доказів і чому вона підвищує переконливість висновку?
6. Що таке робочі документи аудитора і яку функцію вони виконують?
7. Порівняйте функцію робочих документів аудитора й ланцюга зберігання доказів (підрозділ 2.10 модуля 1).
8. Опишіть чотири рівні критичності невідповідностей за результатами аудиту.
9. Чому невідповідність повинна бути класифікована, а не просто зафіксована як факт?
10. З яких основних розділів складається звіт аудиту?
11. Чому звіт аудиту повинен містити не лише перелік проблем, а й позитивні висновки?

12. Що таке план коригувальних дій (CAP) і хто його розробляє?
13. Чому саме аудований суб'єкт, а не аудитор, розробляє план коригувальних дій?
14. Що таке подальший контроль (follow-up) і навіщо він потрібен?
15. Що таке забезпечення якості аудиту і які два його основні механізми?
16. Наведіть приклад помилки на етапі планування аудиту, яка здатна підірвати достовірність усього звіту.

РОЗДІЛ 20. АУДИТ З ВИКОРИСТАННЯМ СКАНЕРІВ БЕЗПЕКИ (ПОГЛИБЛЕНО)



Зміст розділу

- 20.1. Роль автоматизованого сканування в програмі аудиту**
- 20.2. Класи сканерів безпеки**
- 20.3. Планування й авторизація сканування**
- 20.4. Інтерпретація та підтвердження результатів сканування**
- 20.5. Сканування вразливостей проти тестування на проникнення**
- 20.6. Автоматизація регулярного сканування в програмі аудиту**
- 20.7. Комплаєнс-сканування конфігурації**
- 20.8. Обмеження автоматизованого сканування і роль аудитора**
- 20.9. Сканування вебзастосунків (DAST) як спеціалізований клас сканерів**

Практикум до розділу 1 модуля 1 уже познайомив читача зі сканерами вразливостей на рівні базового практичного застосування. Цей розділ повертається до тієї самої категорії інструментів, але вже з погляду аудитора третьої лінії (підрозділ 17.3): не як до інструмента, що адміністратор застосовує до власної інфраструктури, а як до одного з методів збору технічних доказів аудиту (підрозділ 19.3), з усіма супутніми вимогами до авторизації, документування й інтерпретації результату.

20.1. Роль автоматизованого сканування в програмі аудиту

Автоматизоване сканування дає аудиту те, чого не може дати жоден із суто документальних методів, об'єктивний, технічно підтверджений знімок фактичного стану системи в конкретний момент часу, незалежний від того, що стверджує документація чи персонал аудованого суб'єкта. Саме тому технічне тестування (підрозділ 19.3), і сканування безпеки як його найпоширеніша автоматизована форма, є одним із найпереконливіших джерел доказів аудиту: результат сканування важко оспорити посиланням на суб'єктивну інтерпретацію, оскільки він відтворюваний і документований самим інструментом.

Водночас сканування, як буде показано в підрозділі 20.8, ніколи не замінює аудит повністю, а лише постачає йому один із класів доказів, який потребує подальшої інтерпретації кваліфікованим аудитором у контексті решти зібраних матеріалів.

20.2. Класи сканерів безпеки

Розширюючи короткий перелік із практикуму до розділу 1 модуля 1, варто розрізняти три функціонально різні класи сканерів. **Мережеві сканери** (Nmap та подібні) виявляють активні вузли мережі, відкриті порти й запущені служби, відповідаючи передусім на запитання «що взагалі присутнє в мережі» ще до оцінки вразливостей конкретних компонентів, тісно пов'язане з інвентаризацією активів (підрозділ 9.7 модуля 2). **Сканери вразливостей** (Nessus, OpenVAS/Greenbone) зіставляють версії виявленого програмного забезпечення з базами відомих вразливостей CVE (підрозділ 9.6 модуля 2), оцінюючи їх за шкалою CVSS. **Сканери вебзастосунків** (наприклад, OWASP ZAP, Burp Suite) спеціалізуються на прикладному рівні (підрозділ 13.4 модуля 2), активно взаємодіючи із застосунком для виявлення ін'єкцій, міжсайтового скриптингу та інших уразливостей, специфічних саме для логіки вебзастосунку, а не для версій встановленого програмного забезпечення.

Кожен клас відповідає на власне, лише частково перекривне запитання, тому повноцінна програма технічного тестування в межах аудиту, як правило, поєднує всі три, а не обмежується одним інструментом.

20.3. Планування й авторизація сканування

Сканування безпеки, на відміну від пасивного моніторингу (розділ 1 модуля 1), є активним втручанням у роботу мережі й систем: агресивне сканування здатне перевантажити вразливий сервіс чи навіть спричинити його відмову. Тому будь-яке сканування в межах аудиту вимагає формальної **авторизації**: письмового дозволу від відповідальної особи аудованого суб'єкта, що чітко визначає межі сканування (**Rules of Engagement**), дозволений діапазон адрес, часове вікно проведення (типово поза пікові години навантаження), заборонені типи агресивних перевірок (наприклад, тести на відмову в обслуговуванні) і контактну особу на випадок непередбачених наслідків.

Відсутність такої авторизації перетворює технічно ідентичну дію з законного методу збору доказів аудиту на неправомірне втручання в роботу інформаційної системи, юридично й етично невиправдане незалежно від добрих намірів аудитора: межа між авторизованим тестуванням і протиправною дією визначається виключно наявністю формального дозволу, а не технікою самого сканування.

20.4. Інтерпретація та підтвердження результатів сканування

Сирий вивід сканера рідко можна використовувати як готовий доказ аудиту без подальшої обробки. **Дедуплікація** об'єднує повторювані знахідки (той самий вузол, просканий кількома методами, часто породжує кілька записів про одну й ту саму вразливість). **Верифікація** підтверджує, що знайдена вразливість справді експлуатована в конкретному контексті: сканер, орієнтований на широке покриття, схильний повідомляти про потенційні вразливості на основі версії програмного забезпечення, не завжди враховуючи компенсаційні заходи контролю (наприклад, вразливий сервіс, доступний лише з ізольованого сегмента мережі, підрозділ 10.4 модуля 2, становить менший практичний ризик, ніж той самий сервіс на загальнодоступному периметрі).

Практичний результат цього етапу, **пріоритизований перелік підтверджених знахідок**, готовий для формулювання невідповідностей аудиту (підрозділ 19.5), істотно коротший за сирий вивід сканера, але значно переконливіший як доказ, оскільки кожен запис у ньому вже пройшов критичну оцінку аудитора, а не є сліпо перенесеним результатом автоматизованого інструмента.

20.5. Сканування вразливостей проти тестування на проникнення

Підрозділ 17.7 коротко розмежував аудит і тестування на проникнення за метою; тепер розглянемо технічну відмінність двох суміжних форм технічного тестування детальніше. **Сканування вразливостей** є переважно автоматизованим, широким за охопленням і неглибоким за кожною окремою знахідкою процесом: сканер повідомляє про потенційну наявність вразливості на основі версії програмного забезпечення чи конфігурації, але, як правило, не намагається довести можливість її практичної експлуатації. **Тестування на проникнення**, навпаки, є переважно ручним (хоча й з використанням автоматизованих інструментів на окремих етапах), вузьким за охопленням, але суттєво глибшим: кваліфікований фахівець активно намагається використати виявлені вразливості для практичного досягнення конкретної мети (наприклад, отримання доступу до бази даних), імітуючи дії реального зловмисника, за принципом, спорідненим з red team із підрозділу 12.9 модуля 2, хоча зазвичай вузьким за обсягом і формальнішим за методологією.

Практичний висновок для аудиту: сканування вразливостей відповідає на запитання «що потенційно вразливе», а тестування на проникнення, на запитання «що з цього справді можна практично використати проти організації», і саме тому обидва методи взаємно доповнюють одне одного в межах повноцінної технічної перевірки, а не конкурують за одну й ту саму роль.

20.6. Автоматизація регулярного сканування в програмі аудиту

Одноразове сканування, проведене виключно під час конкретної аудиторської перевірки, дає лише знімок стану на момент перевірки: нова вразливість, опублікована наступного дня після завершення аудиту, лишиться непоміченою до наступного циклу. Тому зріла програма аудиту (підрозділ 17.8) поєднує періодичні поглиблені аудиторські перевірки з **безперервним автоматизованим скануванням**, що виконується значно частіше (щотижнево чи навіть щодня для критичних систем), як окрема функція управління вразливостями (підрозділ 9.6 модуля 2), результати якої аудитор використовує як додаткове, вже накопичене джерело доказів під час формальної перевірки, а не розпочинає збір цих даних із нуля.

Таке поєднання ілюструє загальний принцип взаємодії трьох ліній захисту (підрозділ 17.3): регулярне автоматизоване сканування є функцією першої й другої ліній (адміністрування й моніторинг вразливостей), тоді як третя лінія (аудит)

періодично незалежно перевіряє, чи справді ця функція виконується ефективно та своєчасно.

20.7. Комплаєнс-сканування конфігурації

Окремим, вужчим за призначенням класом автоматизованого сканування є **комплаєнс-сканування (configuration compliance scanning)**: на відміну від сканування вразливостей, що шукає відомі технічні недоліки програмного забезпечення, комплаєнс-сканування перевіряє відповідність фактичної конфігурації системи наперед визначеному еталонному профілю безпечного налаштування, найпоширеніше **CIS Benchmarks** (Center for Internet Security) чи галузевим стандартом **DISA STIG** (Security Technical Implementation Guide), обидва вже коротко згадані в практикумі до розділу 1 модуля 1.

Такий еталонний профіль містить сотні конкретних, перевірюваних параметрів (наприклад, «мінімальна довжина пароля не менше 14 символів», «служба Telnet вимкнена», «аудит подій входу в систему увімкнено») для конкретної операційної системи чи застосунку, і саме порівняно з цим профілем, а не з абстрактним поняттям «безпечне налаштування», комплаєнс-сканер формує кількісну оцінку відповідності (наприклад, «87% параметрів профілю CIS Benchmark для Windows Server відповідають вимозі»), придатну для прямого включення у звіт аудиту (підрозділ 19.6) як кількісний, легко відтворюваний доказ.

20.8. Обмеження автоматизованого сканування і роль аудитора

Попри всю технічну цінність, автоматизоване сканування має суттєві структурні обмеження. Сканер перевіряє лише те, що передбачено в його базі сигнатур чи еталонному профілі, і за визначенням не здатний виявити логічну помилку бізнес-процесу (наприклад, процедуру, що технічно коректно налаштована, але організаційно дозволяє обійти принцип поділу обов'язків, підрозділ 10.1 модуля 2), відсутність задокументованого плану реагування на інциденти (підрозділ 10.5 модуля 2), чи невідповідність між задекларованою й фактичною практикою, яку виявляє лише інтерв'ю (підрозділ 19.3).

Тому кінцева відповідальність за формулювання висновку аудиту завжди лишається за кваліфікованим аудитором-людиною, а не за автоматизованим інструментом: сканування безпеки постачає важливий, технічно переконливий, але лише частковий клас доказів, який аудитор зобов'язаний зважити разом з рештою зібраного матеріалу, а не сліпо перенести у фінальний звіт як готовий висновок.

20.9. Сканування вебзастосунків (DAST) як спеціалізований клас сканерів

Загальні сканери вразливостей інфраструктури, розглянуті раніше в цьому розділі, перевіряють переважно відомі, каталогізовані вразливості операційних систем і мережевих служб, але погано пристосовані до виявлення вразливостей логіки власного, унікального коду вебзастосунку, наприклад, помилки авторизації, що дозволяє одному користувачеві переглядати чужі записи, попри те що ні операційна система, ні жодна бібліотека самі по собі не є вразливими. **DAST (Dynamic Application Security Testing)** позначає клас спеціалізованих сканерів, що взаємодіють із запущеним вебзастосунком ззовні, так само як це робив би реальний користувач чи зловмисник, підставляючи в кожне поле вводу й параметр запиту типові шкідливі значення (SQL-ін'єкція, XSS та подібні) і аналізуючи відповідь застосунку на предмет ознак успішної експлуатації.

DAST природно доповнює SAST (Static Application Security Testing, аналіз вихідного коду до запуску, згаданий у контексті DevSecOps-конвеєра в підрозділі 9.9 модуля 2): SAST виявляє потенційно небезпечний патерн коду ще до розгортання, тоді як DAST перевіряє застосунок у реальному, розгорнутому стані, включно з ефектами конфігурації сервера й середовища виконання, які SAST у принципі не бачить. Для аудитора практичний висновок з DAST-сканування цінний саме тим, що він відтворює зовнішню перспективу реального зловмисника, доповнюючи внутрішні перевірки конфігурації (розділ 21) незалежним, емпіричним підтвердженням чи спростуванням захищеності вебзастосунку.

Практикум 20. Комплексне сканування навчального середовища для потреб аудиту

Мета роботи: провести автоматизоване сканування із застосуванням формальної процедури авторизації та подальшої верифікації результатів у ролі аудитора, а не адміністратора.

Хід роботи:

1. скласти формальний документ авторизації сканування (Rules of Engagement) для тестового сегмента мережі: діапазон адрес, часове вікно, заборонені дії, контактна особа;
2. виконати мережеве сканування (виявлення вузлів і портів) та сканування вразливостей тестового сегмента, отриманого раніше в практикумі до розділу 1 модуля 1;

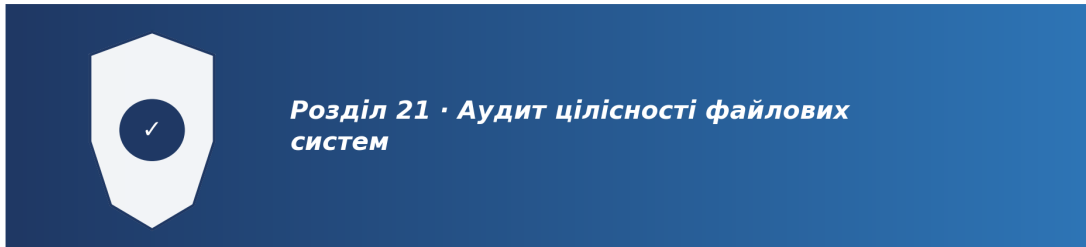
3. виконати комплаєнс-сканування щонайменше однієї системи проти доступного профілю CIS Benchmark чи аналогічного еталона;
4. дедуплікувати й верифікувати отримані знахідки, відзначивши для кожної підтвердженої вразливості наявність чи відсутність компенсаційних заходів контролю;
5. оформити пріоритизований перелік підтверджених знахідок, придатний для включення у звіт аудиту за структурою підрозділу 19.6.

Контрольні питання до розділу 20

1. Чим сканування як метод збору доказів аудиту відрізняється від того самого сканування, виконаного адміністратором для власних потреб?
2. Перелічіть три класи сканерів безпеки та відмінність у запитаннях, на які кожен з них відповідає.
3. Що таке Rules of Engagement і чому вони обов'язкові перед скануванням у межах аудиту?
4. Яка юридична різниця між авторизованим і неавторизованим скануванням тієї самої мережі?
5. Що таке дедуплікація результатів сканування?
6. Що таке верифікація знайденої вразливості і чому вона важлива для якості доказу аудиту?
7. Порівняйте сканування вразливостей і тестування на проникнення за глибиною й охопленням.
8. На яке запитання відповідає сканування вразливостей, а на яке, тестування на проникнення?
9. Чому одноразового сканування під час аудиторської перевірки недостатньо для повноцінного управління вразливостями?
10. Що таке комплаєнс-сканування конфігурації і чим воно відрізняється від сканування вразливостей?
11. Наведіть приклад еталонного профілю для комплаєнс-сканування.
12. Яку кількісну оцінку здатне надати комплаєнс-сканування для звіту аудиту?
13. Назвіть щонайменше два типи проблем, які автоматизоване сканування принципово не здатне виявити.
14. Чому кінцева відповідальність за висновок аудиту завжди лишається за аудитором-людиною?

15. Як пов'язані регулярне автоматизоване сканування (перша/друга лінія) і періодичний аудит (третя лінія) з погляду моделі трьох ліній?

РОЗДІЛ 21. АУДИТ ЦІЛІСНОСТІ ФАЙЛОВИХ СИСТЕМ



Зміст розділу

- 21.1. Поняття цілісності даних і файлової системи***
- 21.2. Принцип роботи File Integrity Monitoring (FIM)***
- 21.3. Інструменти FIM***
- 21.4. Побудова базового профілю файлової системи***
- 21.5. FIM у вимогах стандартів аудиту***
- 21.6. Хибнопозитивні спрацювання FIM і тюнінг***
- 21.7. Цілісність у хмарному й контейнеризованому середовищі***
- 21.8. Цілісність баз даних: контроль на рівні СУБД***

Розділ 20 розглядав аудит вразливостей, тобто пошук слабких місць, якими зловмисник міг би скористатися. Цей розділ ставить інше запитання: як дізнатися, що зловмисник (чи, не менш імовірно, помилка адміністратора) вже змінив щось у системі без санкції. Відповідь на нього дає **аудит цілісності файлових систем**, окрема, вузькоспеціалізована, але практикумом робочої програми прямо передбачена тема.

21.1. Поняття цілісності даних і файлової системи

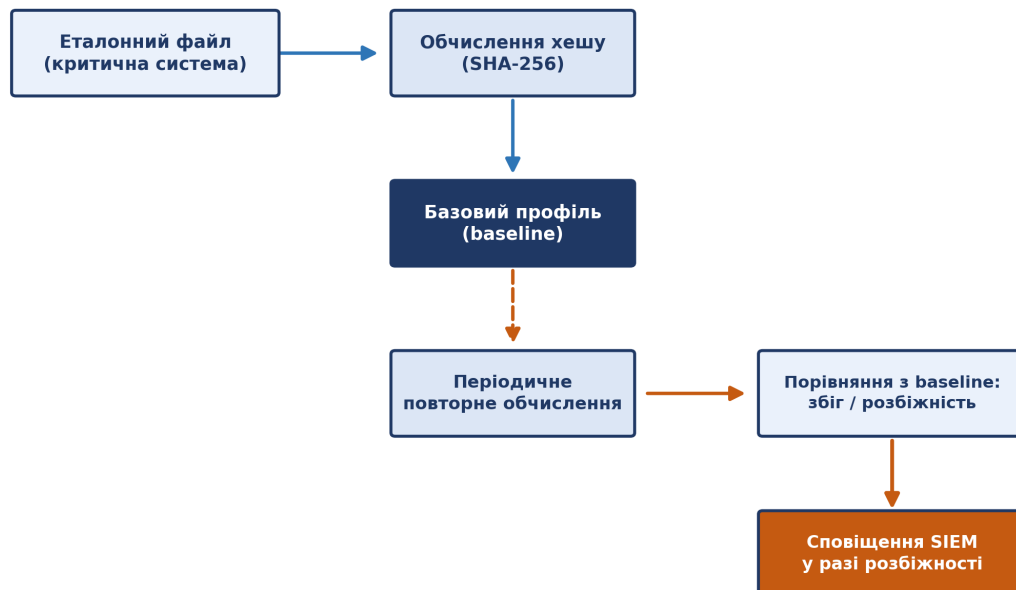
Цілісність (integrity), одна з трьох класичних властивостей інформаційної безпеки поряд із конфіденційністю й доступністю, означає, що дані залишаються точними, повними й незміненими за відсутності санкціонованої на те дії. Стосовно файлової системи це означає, що критичний файл (виконуваний файл системної

служби, конфігураційний файл, бібліотека) містить саме той вміст, який мав містити відповідно до останньої санкціонованої зміни, зафіксованої, зокрема, процесом управління змінами (розділ 9 модуля 2).

Порушення цілісності файлової системи є характерним, хоча далеко не єдиним слідом успішної атаки: зловмисник, що отримав доступ до системи, майже завжди змінює або додає файли, встановлює бекдор, підмінює системну утиліту на шкідливу версію, або змінює конфігураційний файл для приховування власної активності (тема, до якої розділ 22 повертається детальніше). Виявлення саме таких змін, а не запобігання їм (це завдання належить радше до превентивних заходів контролю доступу, розглянутих у модулі 2), і є метою File Integrity Monitoring.

21.2. Принцип роботи File Integrity Monitoring (FIM)

Механізм FIM, зображений на рисунку 21.1, спирається на криптографічні **хеш-функції**: односторонні математичні перетворення, що для будь-якого вхідного файлу обчислюють значення фіксованої довжини (хеш), причому навіть мінімальна зміна вхідного файлу, зміна лише одного біта, призводить до кардинально іншого значення хешу. Це робить хеш зручним, компактним «відбитком» вмісту файлу, аналогічним до цифрового відбитка документа з підрозділу 7.2 модуля 1, застосованого тепер не до контенту DLP-політики, а до самого файлу файлової системи.



*Рис. 21.1. Принцип роботи File Integrity Monitoring
Джерело: розроблено автором*

Практично FIM обчислює хеш (найпоширеніше SHA-256, оскільки застарілі алгоритми MD5 і SHA-1 визнані криптографічно вразливими до колізій) кожного файлу з переліку контрольованих об'єктів у момент, коли система вважається довіреною (**базовий профіль**, підрозділ 21.4), а потім періодично повторює обчислення й порівнює новий хеш із збереженим базовим значенням: збіг підтверджує незмінність файлу, а розбіжність сигналізує про зміну, санкціоновану чи ні, яка потребує подальшого розслідування.

21.3. Інструменти FIM

Історично найвідомішим комерційним рішенням класу FIM є **Tripwire**, що дав назву самому підходу в багатьох текстах з безпеки. Серед відкритих альтернатив поширений **AIDE (Advanced Intrusion Detection Environment)** для Linux-систем, що дозволяє гнучко налаштувати перелік контрольованих файлів і каталогів, а також те, які саме атрибути файлу відстежувати (не лише хеш вмісту, а й права доступу, власника, час модифікації). Дедалі частіше функціональність FIM вбудовується безпосередньо в ширші платформи безпеки: сучасні EDR-рішення (підрозділ 12.8 модуля 2) і хмарні платформи захисту (наприклад, модулі File Integrity Monitoring в складі рішень класу CSPM, підрозділ 14.7 модуля 2) пропонують FIM як одну з багатьох функцій єдиної консолі, а не окремий спеціалізований продукт.

Вибір конкретного інструменту на практиці часто визначається не стільки технічними відмінностями (принцип роботи, описаний у підрозділі 21.2, спільний для всіх), скільки зручністю інтеграції з наявною інфраструктурою моніторингу: FIM, що надсилає сповіщення безпосередньо в SIEM (розділ 3 модуля 1) як ще одне джерело подій, суттєво практичніший за ізольований інструмент з окремою консолою, яку доводиться перевіряти вручну.

21.4. Побудова базового профілю файлової системи

Якість роботи FIM повністю залежить від якості **базового профілю (baseline)**, з яким порівнюється кожен наступний стан системи, тому побудова базового профілю є не одноразовим технічним кроком, а відповідальним організаційним рішенням. Базовий профіль слід будувати лише на системі, стан якої на момент побудови достеменно відомий як довірений (щойно розгорнута й повністю оновлена

система, або система одразу після підтвердженого успішного застосування планового патчу, підрозділ 9.8 модуля 2), інакше FIM ризикує «законсервувати» вже наявну, але ще не виявлену компрометацію як хибний еталон норми.

Вибір самого переліку контрольованих файлів вимагає балансу, добре знайомого з інших контекстів книги (наприклад, вибір джерел журналювання, підрозділ 2.2 модуля 1): контроль абсолютно всіх файлів файлової системи, включно з тимчасовими й часто змінюваними користувацькими документами, породжує величезний обсяг хибнопозитивних сповіщень (підрозділ 21.6) і невиправдане навантаження на систему, тоді як надто вузький перелік ризикує пропустити файл, що справді має значення для цілісності системи. Практика типово обмежує контроль системними виконуваними файлами, бібліотеками, конфігураційними файлами критичних служб і файлами, що прямо згадані у вимогах відповідного стандарту (підрозділ 21.5).

21.5. FIM у вимогах стандартів аудиту

FIM не є суто технічною рекомендацією «доброї практики»: для окремих категорій систем його впровадження прямо вимагається стандартами, розглянутими в розділі 18. **PCI DSS**, згаданий у підрозділі 18.5, містить пряму вимогу до File Integrity Monitoring для систем у межах середовища обробки карткових даних, включно з обов'язковим сповіщенням відповідального персоналу про будь-яку неавторизовану зміну критичних системних файлів. Стандарти на основі NIST SP 800-53 (підрозділ 18.3) також включають відповідний захід контролю в категорії цілісності системи й інформації (System and Information Integrity).

Для аудитора, що перевіряє відповідність організації одному з цих стандартів, наявність і фактична (а не лише декларована) робота FIM є одним із конкретних, легко верифікованих (через технічне тестування, підрозділ 19.3) пунктів чекліста аудиту, і, відповідно, часто саме та невідповідність, яку найпростіше виявити й найважче переконливо заперечити аудованому суб'єкту, оскільки доказ (наявність чи відсутність увімкненого й актуального FIM-агента) є однозначним технічним фактом.

21.6. Хибнопозитивні спрацювання FIM і тюнінг

Легітимні причини зміни файлу, встановлення планового оновлення (підрозділ 9.8 модуля 2), санкціоноване розгортання нової версії застосунку через конвеєр CI/CD (підрозділ 15.4 модуля 2), рутинна ротація журналів, виникають так

само часто, як і несанкціоновані. Без належного тюнінгу FIM генерує ту саму проблему **втоми від сповіщень (alert fatigue)**, що вже неодноразово поставала в книзі стосовно IDS/IPS (підрозділ 12.5 модуля 2) і DLP (підрозділ 7.7 модуля 1): адміністратор, засипаний сповіщеннями про кожне планове оновлення, поступово починає ігнорувати їх усі, включно з тим одним справжнім сигналом компрометації, що загубився серед шуму.

Практичний тюнінг передбачає **інтеграцію з процесом управління змінами (підрозділ 9.5 модуля 2)**: заздалегідь узгоджене вікно обслуговування чи запланований реліз тимчасово переводить контрольовані файли в режим очікуваної зміни, і FIM автоматично оновлює базовий профіль після підтвердженого успішного завершення санкціонованої зміни, замість того щоб щоразу генерувати сповіщення про очікувану, законну модифікацію.

21.7. Цілісність у хмарному й контейнеризованому середовищі

Класична модель FIM, описана в підрозділах 21.2-21.4, спроектована для довготривалого сервера з відносно стабільним набором файлів. Контейнеризоване середовище (розділ 15 модуля 2) руйнує це припущення: контейнер, за визначенням, часто живе лічені хвилини чи години, а перевстановлення файлів усередині нього при кожному новому розгортанні є нормою, а не відхиленням. Тому класична FIM, орієнтована на тривалий моніторинг незмінного файлу на постійному сервері, значною мірою поступається місцем принципу **незмінної інфраструктури (immutable infrastructure)**: замість перевірки того, що файл усередині вже запущеного контейнера не змінився, організація гарантує цілісність через **підписування образів (image signing)**, коли кожен контейнерний образ (підрозділ 15.2 модуля 2) криптографічно підписується довіреним джерелом одразу після складання в конвеєрі CI/CD, а кластер Kubernetes налаштований приймати до розгортання лише підписані образи з довіреного реєстру, тим самим гарантуючи цілісність не перевіркою вже запущеної системи, а недопущенням розгортання нецілісного образу від самого початку.

Цей підхід, превентивний, а не детективний, добре узгоджується із загальним принципом ешелонованого захисту (підрозділ 10.2 модуля 2): для короткоживучих контейнерів рубіж «не допустити нецілісний образ до запуску» практично ефективніший, ніж рубіж «виявити зміну файлу всередині вже запущеного, ефемерного контейнера».

21.8. Цілісність баз даних: контроль на рівні СУБД

Розглянутий у попередніх підрозділах моніторинг цілісності файлів застосовується на рівні файлової системи, проте для системи керування базами даних (СУБД) сама цілісність критичних даних значно частіше порушується не через зміну файлу на диску, а через несанкціоновану зміну запису безпосередньо через SQL-інтерфейс, включно з діями користувача з легітимними обліковими даними, що зловживає наданими правами. **Аудит на рівні СУБД (database activity monitoring, DAM)** доповнює файлову FIM, фіксуючи й аналізуючи самі SQL-операції, а не лише їхній кінцевий ефект на файлах даних, дозволяючи відповісти на питання «хто, коли і яким запитом змінив цей конкретний запис», що недоступне засобам файлової цілісності.

Практична реалізація спирається на вбудовані механізми аудиту самої СУБД (наприклад, PostgreSQL audit extension чи SQL Server Audit) або на спеціалізовані проксі-рішення, що перехоплюють весь трафік до бази даних незалежно від облікового запису, яким виконано запит, і зіставляють кожну зміну з очікуваним патерном роботи прикладного застосунку. Виявлена автоматизованим сканером бази даних вразливість (наприклад, надмірні права облікового запису застосунку, здатного виконувати DDL-команди, яких прикладна логіка насправді не потребує) є типовим предметом такого аудиту й прямо перетинається з принципом найменших привілеїв, розглянутим у підрозділі 10.1 модуля 2.

Практикум 21. Розгортання File Integrity Monitoring для критичного каталогу

Мета роботи: отримати практичні навички налаштування й тестування FIM на прикладі відкритого інструменту.

Хід роботи:

1. встановити відкритий інструмент FIM (наприклад, AIDE) у тестовому середовищі Linux;
2. визначити перелік контрольованих каталогів і файлів (наприклад, системні виконувані файли й конфігураційні файли ключових служб) і побудувати базовий профіль;
3. внести свідому, санкціоновану зміну до одного з контрольованих файлів (імітація планового оновлення) і перевірити спрацювання;
4. внести свідому, несанкціоновану зміну (імітація дії зловмисника, наприклад підміну виконуваного файлу) і порівняти сповіщення з попереднім пунктом;

5. оновити базовий профіль після підтвердження санкціонованої зміни й описати, яким чином така процедура інтегрується з процесом управління змінами (підрозділ 9.5 модуля 2).

Контрольні питання до розділу 21

1. Дайте означення цілісності даних і поясніть, чому це одна з трьох класичних властивостей інформаційної безпеки.
2. Опишіть принцип роботи FIM на основі хеш-функцій (рисунок 21.1).
3. Чому для FIM не рекомендовано застосовувати алгоритми MD5 чи SHA-1?
4. Назвіть відомі інструменти FIM, комерційні й з відкритим кодом.
5. Чому базовий профіль повинен будуватися лише на достовірно довірєній системі?
6. Які ризики пов'язані з надто широким переліком контрольованих файлів у FIM? А з надто вузьким?
7. Який стандарт прямо вимагає впровадження FIM і для яких систем?
8. Чому наявність чи відсутність FIM є зручним, легко верифіковним доказом аудиту?
9. Що спричиняє хибнопозитивні спрацювання FIM?
10. Як інтеграція FIM з процесом управління змінами (розділ 9 модуля 2) знижує кількість хибнопозитивних спрацювань?
11. Чому класична модель FIM погано підходить для контейнеризованого середовища?
12. Що таке підписування образів (image signing) і як воно замінює класичний FIM у Kubernetes?
13. Порівняйте превентивний (image signing) і детективний (класична FIM) підходи до цілісності в контексті ешелонованого захисту.
14. Наведіть приклад санкціонованої й несанкціонованої зміни файлової системи.
15. Чим FIM концептуально схожа на цифровий відбиток документа з DLP (підрозділ 7.2 модуля 1)?

РОЗДІЛ 22. ДОСЛІДЖЕННЯ ЗАСОБІВ ПЕРЕШКОДЖАННЯ АУДИТУ ІНФОРМАЦІЇ В ІТ-СИСТЕМІ



Розділ 22 · Засоби перешкоджання аудиту інформації

Зміст розділу

- 22.1. Поняття протидії аудиту та мотивація зловмисника*
- 22.2. Очищення й підробка журналів*
- 22.3. Приховування даних: стеганографія, альтернативні потоки, timestomping*
- 22.4. Ухилення від сканерів і засобів аудиту*
- 22.5. Захисні заходи проти протидії аудиту*
- 22.6. Виявлення ознак протидії аудиту під час перевірки*
- 22.7. Юридичні наслідки навмисної протидії аудиту*
- 22.8. Протидія аудиту в хмарному середовищі: видалення CloudTrail/Activity Logs*

Розділи 17-21 виходили з припущення, що аудований суб'єкт, включно з потенційним зловмисником, що вже проник у систему, не докладає активних зусиль, щоб обдурити аудитора. Це припущення справедливе далеко не завжди: досвідчений зловмисник, як і, зрідка, недобросовісний співробітник, зацікавлений у прихованні власних слідів, свідомо застосовує техніки, розроблені спеціально для протидії виявленню й аудиту. Розуміння цих технік необхідне аудитору не для їх застосування, а для того, щоб розпізнати їхні ознаки під час перевірки.

22.1. Поняття протидії аудиту та мотивація зловмисника

Протидія аудиту (anti-forensics у ширшому сенсі, застосованому до аудиту) — сукупність технік, спрямованих на приховування, спотворення чи знищення доказів, які інакше дозволили б аудитору чи слідчому встановити факт і масштаб порушення. Мотивація для застосування таких технік поділяється на дві принципово різні категорії: **зовнішній зловмисник**, що прагне максимально подовжити час непоміченої присутності в скомпрометованій системі (аналогічно до техніки латерального руху з підрозділу 6.8 модуля 1, приховування слідів продовжує, а не замінює саму атаку), і **внутрішній суб'єкт** (недобросовісний співробітник чи навіть адміністратор, що приховує власну помилку чи порушення),

зацікавлений уникнути виявлення власної дії під час майбутнього аудиту чи розслідування.

Незалежно від мотивації, розуміння конкретних технік протидії аудиту є необхідною частиною компетенції аудитора третьої лінії (підрозділ 17.3): аудитор, що не знає, як виглядають типові сліди приховування доказів, ризикує прийняти штучно «чисту» систему за дійсно чистою.

22.2. Очищення й підробка журналів

Найпряміша й водночас найпоширеніша техніка протидії аудиту спрямована безпосередньо на журнали, головне джерело доказів, розглянуте в розділі 2 модуля 1. **Очищення журналів (log clearing)** повністю видаляє записи журналу подій за певний період чи весь журнал цілком: у Windows команда очищення журналу подій сама по собі генерує характерний запис про факт очищення, що, за іронією, стає першою ознакою протидії аудиту, якщо цей запис вчасно не помічено чи не видалено разом з рештою. **Вибіркове видалення (log editing)**, технічно складніша техніка, видаляє чи змінює лише конкретні записи, що стосуються дій зловмисника, залишаючи решту журналу незміненою, щоб уникнути очевидної підозри, яку викликає повністю порожній журнал за певний період.

Саме проти цього класу технік спрямовані заходи захисту журналів, уже розглянуті в підрозділі 2.6 модуля 1: модель WORM (Write Once Read Many) і централізоване журналювання (розділ 2 модуля 1), коли копія журналу негайно передається на віддалений сервер збору, недоступний для зміни з боку скомпрометованого джерела, роблять локальне очищення журналу практично марним, оскільки достовірна копія вже існує поза досяжністю зловмисника ще до спроби її знищити.

22.3. Приховування даних: стеганографія, альтернативні потоки, timestomping

Поряд із прямим видаленням доказів зловмисник може приховувати їх, залишаючи фізично присутніми, але непомітними для стандартної перевірки. **Стеганографія** приховує довільні дані (наприклад, викрадену інформацію перед ексфільтрацією чи навіть шкідливий код) усередині файлу іншого, нешкідливого на вигляд формату (зображення, аудіофайл), таким чином, що зовнішній вигляд і навіть розмір файлу-носія змінюються незначно чи не змінюються взагалі, що суттєво ускладнює виявлення звичайним переглядом файлової системи. **Альтернативні потоки даних (Alternate Data Streams, ADS)**, специфічна для файлової системи

NTFS можливість, дозволяють прикріпити додаткові, приховані від стандартного перегляду дані до звичайного файлу без зміни його видимого розміру чи вмісту в стандартному потоці. **Timestomping** змінює часові мітки файлу (час створення, останньої модифікації, останнього доступу) на штучні значення, що імітують «старий», давно існуючий файл, спеціально для введення в оману аналітика, що будує хронологію інциденту за часовими мітками, тим самим прямо атакуючи достовірність доказів, розглянутих у підрозділі 19.3.

Спільна риса всіх трьох технік: вони не знищують доказ, а маскують його присутність чи спотворюють його метадані, тому протидія їм спирається не на резервне копіювання (як для очищення журналів), а на **незалежну від метаданих файлу верифікацію** (наприклад, порівняння хешу FIM з розділу 21, який лишається достовірним індикатором зміни вмісту незалежно від того, які часові мітки штучно виставлено).

22.4. Ухилення від сканерів і засобів аудиту

Техніки ухилення від систем виявлення вторгнень, уже розглянуті в підрозділі 12.6 модуля 2 (фрагментація, кодування, шифрування), застосовні й до автоматизованого сканування в межах аудиту (розділ 20): зловмисник чи недобросовісний адміністратор, що знає про заплановане сканування (наприклад, за витоком інформації про графік аудиту, підрозділ 19.2), може тимчасово приховати шкідливий процес або тимчасово вимкнути агента FIM (підрозділ 21.3) саме на період сканування, а потім відновити початковий стан одразу після його завершення.

Ця техніка прямо демонструє практичну цінність принципу, згаданого в підрозділі 19.2: конфіденційність точного графіка сканування (а не лише самого факту його проведення) є частиною ефективності аудиту, а несподіване, позапланове (surprise) технічне тестування, яке аудований суб'єкт не може підготуватися «пережити» тимчасово, є визнаним, хоча й не завжди практично здійсненим через організаційні обмеження, методом протидії саме такому типу маскування.

22.5. Захисні заходи проти протидії аудиту

Ефективна протидія технікам із підрозділів 22.2-22.4 спирається переважно на заходи, уже введені в попередніх розділах книги, застосовані тепер свідомо саме з міркувань стійкості до протидії аудиту, а не лише загальної гігієни безпеки. **Незмінне централізоване журналювання** (WORM, підрозділ 2.6 модуля 1) робить локальне очищення журналів марним. **FIM** (розділ 21) виявляє зміни файлів

незалежно від маніпуляцій з часовими мітками. **Синхронізація часу через захищений NTP** (підрозділ 11.7 модуля 2) робить timestamping (підрозділ 22.3) помітним при зіставленні з незалежними, централізованими мітками часу журналів. **Розділення обов'язків і принцип найменших привілеїв** (підрозділ 9.4 модуля 2) обмежують саме коло осіб, технічно здатних виконати очищення журналів чи вимкнути FIM-агента, а журналювання підвищення прав (той самий підрозділ 9.4) фіксує сам факт спроби такої дії.

Практичний висновок: не існує єдиного «засобу проти протидії аудиту», а комплексне поєднання вже розглянутих заходів контролю різних розділів книги, застосованих узгоджено, суттєво підвищує вартість і ризик для зловмисника, що намагається приховати сліди своєї присутності.

22.6. Виявлення ознак протидії аудиту під час перевірки

Досвідчений аудитор навчається розпізнавати непрямі ознаки того, що докази могли бути навмисно спотворені, навіть якщо пряме підтвердження факту маніпуляції відсутнє. До типових «червоних прапорців» належать: підозріло рівномірна, «занадто чиста» історія журналу без жодного інциденту чи аномалії за тривалий період, коли решта доступних доказів (наприклад, дані сканування вразливостей чи скарги користувачів) вказують на протилежне; наявність записів про очищення журналу (підрозділ 22.2) саме перед запланованим періодом аудиту; часові мітки файлів, що суперечать логічній послідовності подій (наприклад, файл конфігурації, «модифікований» раніше за дату встановлення самої системи); і невідповідність між результатами різних, незалежних джерел доказів (тріангуляція з підрозділу 19.3), коли, наприклад, журнал стверджує одне, а незалежна копія того самого журналу з централізованого сховища (підрозділ 2.4 модуля 1) стверджує інше.

Виявлення такої розбіжності саме по собі не доводить зловмисного наміру (причиною може бути й технічна помилка), але завжди є достатньою підставою для поглибленого розслідування, що виходить за межі стандартного обсягу планового аудиту й, залежно від масштабу, може перейти в окреме розслідування інциденту.

22.7. Юридичні наслідки навмисної протидії аудиту

Навмисне знищення чи підробка доказів у процесі, що вже має ознаки офіційного розслідування чи аудиту, як правило, тягне юридичні наслідки, окремі й додаткові до наслідків первинного порушення, яке ці дії намагалися приховати: у

багатьох юрисдикціях перешкоджання розслідуванню чи фальсифікація доказів кваліфікується як окреме правопорушення, незалежно від результату основного розслідування. Для співробітника організації навмисне знищення журналів чи маніпуляція системними файлами з метою приховування власних дій, як правило, є грубим порушенням внутрішньої політики (підрозділ 9.2 модуля 2) і підставою для негайного відсторонення й дисциплінарного провадження, незалежно від подальшого юридичного переслідування.

Практичний висновок для аудитора: виявлення ознак навмисної протидії аудиту (підрозділ 22.6) саме тому кваліфікується не як звичайна невідповідність (підрозділ 19.5), а найчастіше одразу як критична, з негайним інформуванням керівництва й, за потреби, залученням юридичної служби чи, у випадках, що підпадають під законодавчо визначені критерії, CERT-UA (підрозділ 10.5 модуля 2), ще до завершення повного циклу аудиторської перевірки.

22.8. Протидія аудиту в хмарному середовищі: видалення CloudTrail/Activity Logs

Розглянуті раніше техніки протидії (очищення локальних журналів, підрозділ 22.2) мають прямий хмарний аналог, специфічний саме для інфраструктури, розглянутої в розділі 14 модуля 2. Зловмисник, що отримав доступ до облікового запису з надмірними правами в хмарному середовищі, здатний не просто змінити конфігурацію, а вимкнути чи вибірково видалити сам механізм аудиторського журналювання хмарного провайдера, наприклад **AWS CloudTrail** чи **Azure Activity Log**, тим самим приховуючи власні подальші дії від будь-якого наступного розслідування так само ефективно, як очищення `auth.log` приховує сліди на локальному сервері.

Ключова відмінність хмарного середовища на користь захисту полягає в тому, що сама подія вимкнення чи зміни налаштувань журналювання, як правило, фіксується на рівні платформи провайдера, поза межами контролю облікового запису клієнта, і не може бути видалена тим самим зловмисником, який щойно вимкнув журналювання: провайдер веде незалежний, невидимий для клієнта запис адміністративних дій над самою платформою. З огляду на це, найкращою практикою захисту є експорт хмарних журналів аудиту в незмінне (immutable), окреме сховище чи в центральний SIEM (розділ 3 модуля 1) у реальному часі, а також налаштування сповіщення, що спрацьовує негайно в разі спроби вимкнення чи зміни самого механізму журналювання, перетворюючи спробу протидії аудиту на подію найвищого пріоритету, а не на непомічену прогалину в доказовій базі.

Практикум 22. Виявлення слідів протидії аудиту в навчальному середовищі

Мета роботи: розпізнати технічні ознаки маніпуляції доказами на прикладі контрольованого навчального сценарію.

Хід роботи:

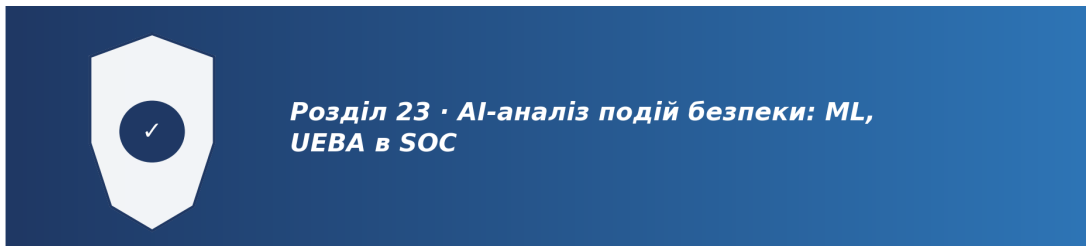
1. у тестовому середовищі виконати санкціоноване очищення частини журналу подій та зафіксувати запис, що сам механізм очищення генерує в журналі;
2. свідомо змінити часові мітки тестового файлу (timestomping) і порівняти видимі метадані файлу з незалежно збереженим хешем FIM (розділ 21) того самого файлу;
3. зіставити локальну копію журналу з централізованою копією, попередньо переданою на віддалений сервер збору (підрозділ 2.4 модуля 1), і зафіксувати розбіжність, якщо вона є;
4. скласти короткий перелік «червоних прапорців», виявлених у ході вправи, за категоріями з підрозділу 22.6;
5. сформулювати, які саме заходи захисту з підрозділу 22.5 унеможливили б чи істотно ускладнили кожен зі змодельованих технік протидії.

Контрольні питання до розділу 22

1. Дайте означення протидії аудиту та поясніть дві категорії мотивації її застосування.
2. Чим очищення журналу відрізняється від вибіркового редагування журналу?
3. Чому спроба очищення журналу подій Windows сама по собі часто залишає слід?
4. Що таке стеганографія і як вона застосовується для приховування даних у файловій системі?
5. Що таке альтернативні потоки даних (ADS) і в якій файловій системі ця можливість присутня?
6. Що таке timestomping і яку саме властивість доказів він атакує?
7. Чим протидія скануванню в межах аудиту (підрозділ 22.4) технічно споріднена з ухиленням від IDS/IPS (підрозділ 12.6 модуля 2)?
8. Чому конфіденційність точного графіка сканування має значення для ефективності аудиту?
9. Перелічіть щонайменше чотири заходи захисту, що ускладнюють протидію аудиту, і поясніть механізм дії кожного.

10. Чому FIM лишається ефективним індикатором навіть за застосування timestamping?
11. Назвіть щонайменше три «червоні прапорці», що можуть свідчити про протидію аудиту.
12. Чому невідповідність між локальною й централізованою копією журналу є вагомою підставою для поглибленого розслідування?
13. Чи кваліфікують знайдені ознаки протидії аудиту як звичайну невідповідність? Обґрунтуйте відповідь.
14. Які юридичні наслідки може мати навмисне знищення доказів у процесі офіційного розслідування?
15. Чому знищення журналу як таке часто тягне окремі юридичні наслідки, незалежно від наслідків первинного порушення, яке воно намагалося приховати?

РОЗДІЛ 23. АІ-АНАЛІЗ ПОДІЙ БЕЗПЕКИ: МАШИННЕ НАВЧАННЯ, UEBA В SIEM/SOC



Зміст розділу

- 23.1. Типи задач машинного навчання, застосовні до аналізу подій безпеки**
- 23.2. UEBA: поглиблений механізм побудови базової поведінки**
- 23.3. Конвеєр обробки подій: від сирих даних до пріоритизованого інциденту**
- 23.4. Обмеження моделей: дрейф, змагальне машинне навчання, надмірна довіра**
- 23.5. Генеративний ШІ та великі мовні моделі в SOC**
- 23.6. Автономні АІ-агенти для виявлення інцидентів**
- 23.7. Оцінювання якості АІ-аналізу: точність, повнота, F1**
- 23.8. Етичні та регуляторні аспекти застосування ШІ в безпеці**
- 23.9. SOAR і автоматизоване реагування на основі АІ-пріоритизації**

Обсяг подій, які генерують сучасні SIEM (розділ 3), системи виявлення вторгнень (розділ 12 модуля 2) і хмарна інфраструктура (розділи 14-15 модуля 2), давно перевищив можливості суто ручного аналізу чи навіть класичних кореляційних правил (підрозділ 3.8 модуля 1). Машинне навчання й поведінкова аналітика UEBA, вперше згадані в підрозділі 8.7 модуля 1 як складова частина сучасного SIEM, розглядаються в цьому розділі поглиблено як самостійний методологічний інструмент аудиту й моніторингу.

23.1. Типи задач машинного навчання, застосовні до аналізу подій безпеки

У контексті аналізу подій безпеки застосовуються три принципово різні типи задач машинного навчання. **Навчання з учителем (supervised learning)** використовує історичний масив подій, для кожної з яких заздалегідь відомий правильний результат (наприклад, «зловмисне» чи «легітимне»), і будує модель, здатну класифікувати нові, ще не позначені події за аналогією; ключове обмеження цього підходу, розмір і якість позначеного (labeled) навчального набору, часто є вузьким місцем, оскільки підтверджені інциденти безпеки трапляються значно рідше, ніж легітимний трафік. **Навчання без учителя (unsupervised learning)** не потребує заздалегідь позначених прикладів і натомість шукає природні групи (кластери) чи аномалії, що суттєво відхиляються від основної маси спостережень: саме цей підхід лежить в основі UEBA (підрозділ 23.2), оскільки більшість аномалій поведінки не має заздалегідь відомого «правильного» позначення. **Навчання з підкріпленням (reinforcement learning)**, третій, менш поширений у безпеці підхід, навчає модель через послідовність спроб і винагород за правильні дії; у контексті SOC застосовується здебільшого експериментально, наприклад, для автоматичного налаштування порогів кореляційних правил (підрозділ 3.8 модуля 1) з мінімізацією хибних спрацювань.

Практичний висновок для аудитора: розуміння того, який саме тип моделі використовує конкретний продукт, безпосередньо визначає, які запитання ставити постачальнику щодо якості навчального набору даних і періодичності перенавчання моделі.

23.2. UEBA: поглиблений механізм побудови базової поведінки

Принцип базової поведінки (baseline), уже введений на рівні концепції в підрозділі 8.7 модуля 1, технічно реалізується через побудову багатовимірного профілю для кожного суб'єкта (користувача, хоста, сервісного облікового запису):

профіль охоплює типовий час активності, звичний перелік ресурсів доступу, типовий обсяг переданих даних, географію підключень і послідовність дій у межах сесії. Модель UEBA обчислює **оцінку ризику (risk score)** для кожної нової дії суб'єкта як функцію відхилення від його власного історичного профілю, а не від якогось універсального «нормального» еталона: підвищений о 3-й ночі обсяг завантаження файлів є нормою для адміністратора нічної зміни й аномалією для бухгалтера, що завжди працює з 9 до 18.

Важлива методологічна деталь, часто недооцінювана під час впровадження: період початкового навчання моделі (як правило, від 30 до 90 днів спостереження) є вразливим вікном, протягом якого система накопичує базовий профіль, і зловмисник, що вже присутній у мережі саме в цей період (наприклад, через раніше не виявлений компрометований обліковий запис), ризикує бути «вивченим» моделлю як частина нормальної поведінки цього суб'єкта, що є прямим практичним наслідком, який аудитор повинен враховувати під час оцінювання ефективності недавно впровадженої UEBA-системи.

23.3. Конвеєр обробки подій: від сирих даних до пріоритизованого інциденту

Рисунок 23.1 ілюструє типовий конвеєр (pipeline) обробки подій безпеки засобами машинного навчання: сирі події з SIEM, EDR і NetFlow (розділи 3, 12 модуля 2, 6) проходять етап **витягування ознак (feature engineering)**, на якому необроблені записи перетворюються на числові чи категоріальні характеристики, придатні для моделі (наприклад, кількість унікальних хостів, до яких звертався користувач за годину, чи ентропія імені процесу); отримані ознаки подаються на навчену модель, яка обчислює оцінку ризику чи ймовірність аномалії; результат передається аналітику вже у вигляді пріоритизованого інциденту, а не сирого потоку подій.

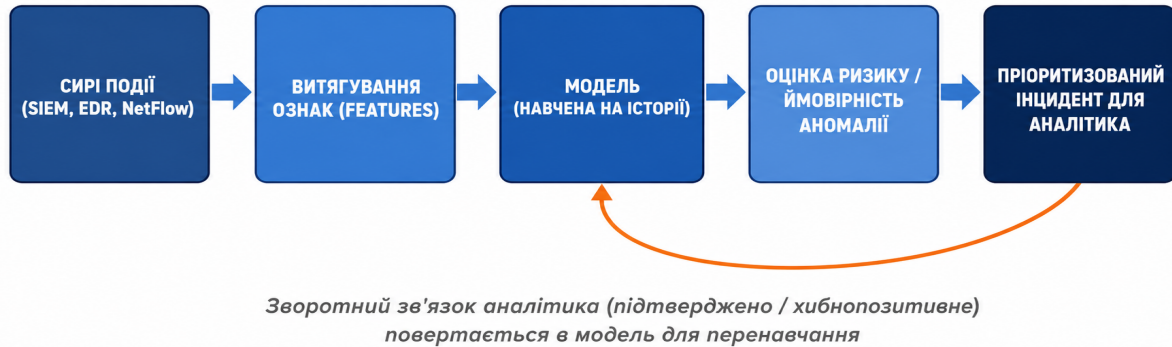


Рис. 23.1. Конвеєр машинного навчання для аналізу подій безпеки

Джерело: розроблено автором.

Ключовий, часто відсутній у спрощених описах елемент конвеєра, це **контур зворотного зв'язку**: підтвердження чи спростування аналітиком кожного згенерованого інциденту (розмічання його як істинно позитивного чи хибнопозитивного) повертається в модель і використовується для періодичного перенавчання, що поступово підвищує точність системи за умови дисциплінованого, послідовного виконання цього кроку командою SOC.

23.4. Обмеження моделей: дрейф, змагальне машинне навчання, надмірна довіра

Жодна модель машинного навчання не є статично точною назавжди. **Дрейф моделі (model drift)** виникає тоді, коли реальне середовище поступово змінюється (нові співробітники, нові сервіси, нова інфраструктура), а модель, навчена на застарілому знімку цього середовища, поступово втрачає точність, якщо її регулярно не перенавчати на свіжих даних. **Змагальне машинне навчання (adversarial ML)** описує клас технік, коли зломисник свідомо конструює свої дії так, щоб залишатися під поріг чутливості моделі, поступово, невеликими кроками, зсуваючи «нормальну» поведінку скомпрометованого облікового запису в бажаному напрямку, технічно споріднено з протидією аудиту з розділу 22, застосованою тепер саме проти моделі, а не проти журналу чи файлової системи. **Надмірна довіра до автоматизованого результату (automation bias)**, суто людський, а не технічний фактор ризику, виникає, коли аналітик SOC починає сприймати оцінку ризику моделі як остаточний вердикт, а не як допоміжний сигнал, що потребує людської

верифікації, особливо в межах вибіркового аудиту (розділ 19), де саме людське судження, а не автоматизований бал, лишається юридично значущим висновком.

Практичний наслідок для аудитора: оцінювання ефективності системи AI-аналізу обов'язково включає перевірку того, чи існує в організації регулярний, задокументований процес перенавчання моделі й чи зберігається людський нагляд (human-in-the-loop) над остаточними рішеннями, а не лише формальний факт наявності самої технології.

23.5. Генеративний ШІ та великі мовні моделі в SOC

Окремим, відносно новим напрямом застосування штучного інтелекту в SOC є генеративні мовні моделі (LLM), що використовуються не для класифікації подій (підрозділи 23.1-23.3), а для допоміжних, текстових задач: автоматичне складання чорнового резюме інциденту природною мовою на основі сирих технічних даних, пришвидшене формулювання рекомендацій з усунення вразливості для менш досвідченого аналітика, і допомога у формулюванні пошукових запитів мовою SIEM (KQL, підрозділ 4.5, чи SPL, підрозділ 5.3) на основі опису задачі природною мовою. Ключове застереження, яке аудитор повинен враховувати: **генеративні моделі схильні до «галюцинацій»**, тобто генерації правдоподібного, але фактично неточного тексту, тому будь-який висновок, сформований LLM, потребує обов'язкової верифікації аналітиком-людиною перед використанням у формальному звіті чи як підстави для реагування, а не сприйняття напряму як достовірного джерела факту.

23.6. Автономні AI-агенти для виявлення інцидентів

Розвитком підходів із підрозділів 23.1-23.5 є AI-агенти, спроектовані для напівавтономного виявлення мережевих інцидентів і некоректних конфігурацій без постійного втручання аналітика на кожному кроці аналізу. Прикладом такого дослідження є робота українських авторів, присвячена застосуванню AI-агентів саме для автоматизованого виявлення мережевих інцидентів і помилок конфігурації: Rudin D., Vasylenko V., Grynkevych G. AI Agents for Automated Network Incident Detection and Misconfiguration Identification. International Scientific Journal «Internauka». 2025. № 12. У цій роботі AI-агент поєднує аналіз мережевого трафіку й конфігураційних даних із моделлю прийняття рішень, що дозволяє автоматично класифікувати виявлену аномалію як імовірний інцидент безпеки або як помилку конфігурації, які на практиці часто важко розрізнити без додаткового контексту

(порівняйте з підрозділом 3.8 модуля 1 щодо джерел хибнопозитивних спрацювань SIEM).

Принциповий методологічний висновок цього й подібних досліджень для SOC-практики: автономність AI-агента доцільно впроваджувати поступово, спершу обмежуючи його роль виявленням і пріоритизацією (аналогічно до конвеєра з підрозділу 23.3), і лише згодом, за підтвердженої точності на історичних даних конкретної організації, розширювати до напівавтоматичних дій реагування, завжди залишаючи людині остаточне рішення щодо критичних, потенційно незворотних кроків (наприклад, ізоляції виробничого хоста).

23.7. Оцінювання якості AI-аналізу: точність, повнота, F1

Ефективність будь-якої моделі машинного навчання в контексті безпеки оцінюється не інтуїтивно, а формальними метриками. **Точність (precision)** показує, яка частка сигналів, позначених моделлю як інцидент, справді є істинними інцидентами (низька точність означає багато хибнопозитивних спрацювань і, як наслідок, втому від сповіщень, підрозділ 3.8 модуля 1); **повнота (recall)** показує, яку частку всіх реальних інцидентів модель дійсно виявила (низька повнота означає пропущені реальні атаки, найнебезпечніший тип помилки для безпеки); **F1-міра**, гармонічне середнє точності й повноти, дає єдиний зведений показник, зручний для порівняння різних моделей чи версій однієї моделі між собою.

Для аудитора, що оцінює впроваджену систему AI-аналізу (розділ 20 у поєднанні з цим розділом), критично важливо вимагати від відповідального підрозділу організації не суб'єктивну оцінку «модель працює добре», а конкретні, виміряні на реальних, історичних даних значення точності й повноти, з поясненням, на якому саме тестовому наборі ці значення обчислено, і чи цей набір репрезентативний для поточного стану інфраструктури організації.

23.8. Етичні та регуляторні аспекти застосування ШІ в безпеці

Застосування машинного навчання для аналізу поведінки користувачів (UEBA, підрозділ 23.2) неминуче зачіпає питання приватності: система, що будує детальний поведінковий профіль кожного співробітника, за своєю природою обробляє персональні дані в значному обсязі, тому впровадження UEBA підпадає під ті самі вимоги захисту персональних даних, що розглядалися стосовно DLP у підрозділі 7.8 модуля 1, і потребує аналогічного балансу між ефективністю виявлення загроз і правом працівника на приватність. Окремим етичним і

практичним ризиком є **алгоритмічна упередженість (bias)**: модель, навчена на історичних даних, що відображають, наприклад, нерівномірний розподіл посад чи графіків роботи в організації, може систематично, хоч і ненавмисно, частіше позначати як «підозрілі» дії певних категорій співробітників, що потребує періодичної, свідомої перевірки моделі саме на предмет такої упередженості, а не лише на предмет технічної точності.

Практичний висновок для аудитора: оцінка впровадженої системи AI-аналізу подій безпеки в межах комплексного аудиту (розділ 24) повинна охоплювати не лише технічну ефективність (підрозділ 23.7), а й відповідність застосування системи вимогам законодавства про захист персональних даних і внутрішній політиці етичного використання автоматизованого прийняття рішень, якщо така політика в організації формалізована.

23.9. SOAR і автоматизоване реагування на основі AI-пріоритизації

Модель виявлення аномалій, розглянута в підрозділах 23.2-23.5, сама по собі лише формує пріоритизований список підозрілих подій; без механізму, що перетворює цей список на дію, аналітик SOC (підрозділ 24.5) все одно залишається перевантаженим обсягом сповіщень (alert fatigue), навіть якщо ці сповіщення тепер якісніше відсортовані за ризиком моделлю UEBA. **SOAR (Security Orchestration, Automation and Response)** доповнює AI-аналіз рівнем автоматизованого реагування: платформа SOAR приймає пріоритизовану подію від SIEM чи моделі UEBA і автоматично виконує заздалегідь визначений сценарій реагування (playbook), наприклад тимчасову ізоляцію хоста, блокування облікового запису чи збагачення події додатковим контекстом з Threat Intelligence (підрозділ 8 модуля 1), без очікування ручного втручання аналітика на кожному кроці.

Поєднання AI-пріоритизації й SOAR дозволяє SOC зосередити обмежений людський ресурс саме на подіях, що дійсно потребують експертного судження, автоматично й миттєво відпрацьовуючи типові, добре зрозумілі сценарії реагування (наприклад, підтверджений збіг із сигнатурою відомого шкідливого програмного забезпечення) без затримки на очікування чергового аналітика. Водночас надмірна довіра до повністю автоматизованого реагування без контрольної точки людини (human in the loop) для нетипових чи високовпливових дій, таких як ізоляція критичного виробничого сервера, сама по собі стає операційним ризиком, тому зрілі впровадження SOAR, як правило, залишають найбільш чутливі кроки playbook на затвердження аналітиком, автоматизуючи лише рутинні, добре перевірені дії.

Практикум 23. Оцінювання якості моделі виявлення аномалій на навчальному наборі даних

Мета роботи: розрахувати й проінтерпретувати метрики точності та повноти для спрощеної моделі виявлення аномалій.

Хід роботи:

1. отримати навчальний набір даних із позначеними подіями (частина яких заздалегідь позначена як «інцидент», частина як «легітимна подія»);
2. застосувати просте порогове правило оцінки ризику (наприклад, за кількістю невдалих спроб автентифікації за одиницю часу) як спрощену модель виявлення;
3. порівняти результат роботи правила з еталонною розміткою й побудувати таблицю істинно позитивних, хибнопозитивних, істинно негативних і хибнонегативних результатів;
4. обчислити точність, повноту та F1-міру для отриманого правила;
5. змінити поріг спрацювання правила і повторно обчислити ті самі метрики, зафіксувавши компроміс між точністю та повнотою;
6. сформулювати висновок про те, який поріг доцільніший для організації з обмеженим штатом аналітиків SOC, і обґрунтувати відповідь.

Контрольні питання до розділу 23

1. Назвіть три типи задач машинного навчання та наведіть приклад застосування кожного до аналізу подій безпеки.
2. Чому навчання без учителя є базовим підходом для UEBA?
3. Що таке базовий профіль поведінки суб'єкта і які параметри він охоплює?
4. Чому початковий період навчання моделі UEBA є вразливим вікном з погляду безпеки?
5. Опишіть послідовність етапів типового конвеєра обробки подій засобами машинного навчання.
6. Яку роль відіграє контур зворотного зв'язку аналітика в підвищенні точності моделі?
7. Що таке дрейф моделі і чому він вимагає регулярного перенавчання?
8. Що таке змагальне машинне навчання і чим воно споріднене з протидією аудиту з розділу 22?

9. Що таке надмірна довіра до автоматизованого результату (automation bias) і чому вона небезпечна саме в контексті аудиту?
10. Для яких задач у SOC застосовуються генеративні мовні моделі (LLM)?
11. Чому результати, згенеровані LLM, потребують обов'язкової верифікації аналітиком?
12. Яку роль відіграють автономні AI-агенти для виявлення мережових інцидентів і помилок конфігурації, за прикладом дослідження Rudin, Vasylenko, Grynkevych (2025)?
13. Чому доцільно впроваджувати автономність AI-агента поступово, а не одразу повністю?
14. Дайте означення точності (precision) та повноти (recall) і поясніть різницю між ними.
15. Чому низька повнота є найнебезпечнішим типом помилки саме для задач безпеки?
16. Які питання приватності виникають при впровадженні UEBA?
17. Що таке алгоритмічна упередженість моделі і як її можна виявити?

РОЗДІЛ 24. КОМПЛЕКСНИЙ ПРАКТИКУМ: ПОБУДОВА SOC-ПРОЦЕСУ МОНІТОРИНГУ, АУДИТУ ТА РЕАГУВАННЯ



Зміст розділу

- 24.1. Карта підручника: як розділи 1-23 складаються в єдиний SOC-процес***
- 24.2. Модель зрілості SOC***
- 24.3. Проектування SOC для гіпотетичної організації: вихідні умови***
- 24.4. Вибір технологічного стеку та обґрунтування рішень***
- 24.5. Робочий процес обробки інциденту від виявлення до закриття***
- 24.6. Панель ключових показників ефективності SOC***

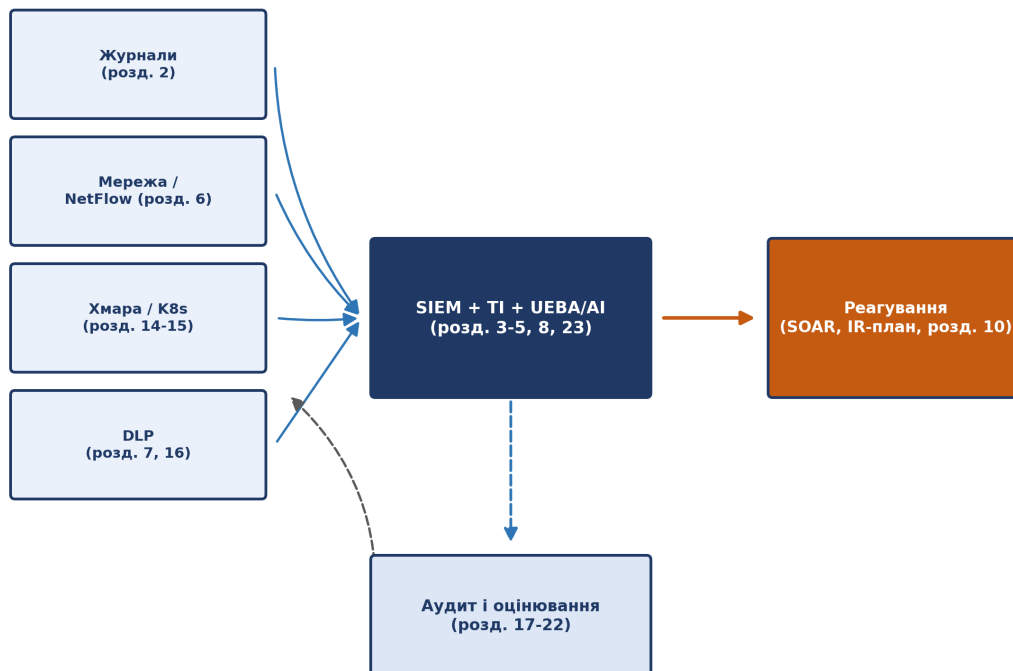
24.7. Перспективи розвитку курсу та подальшого вдосконалення SOC

24.8. Юридична та етична відповідальність SOC-аналітика

Цей завершальний розділ не вводить нового теоретичного матеріалу, а синтезує зміст усіх трьох модулів у єдину, цілісну картину функціонування Центру моніторингу безпеки (SOC), і завершується комплексним практикумом, що вимагає застосування знань і навичок з усього курсу одночасно.

24.1. Карта підручника: як розділи 1-23 складаються в єдиний SOC-процес

Рисунок 24.1 узагальнює архітектуру, що поступово розкривалася протягом усього підручника. Джерела подій, журнали (розділ 2), мережевий трафік (розділ 6), хмарна й контейнеризована інфраструктура (розділи 14-15 модуля 2) і канали DLP (розділи 7, 16), надходять до центрального ядра, що поєднує SIEM (розділ 3), Threat Intelligence (розділ 8) і UEBA/AI-аналіз (розділ 23). Результат аналізу передається до контуру реагування (адміністрування й політики безпеки, розділ 9-13 модуля 2), а весь контур регулярно перевіряється незалежним аудитом (розділи 17-22), що формує рекомендації для його вдосконалення, тим самим замикаючи цикл.



Аудит перевіряє ефективність усього контуру й формує рекомендації для його вдосконалення

Рис. 24.1. Інтегрована архітектура SOC (капстоун)

Джерело: розроблено автором.

Ця схема прямо ілюструє центральну тезу підручника, вже сформульовану на початку модуля 1: моніторинг, адміністрування (захист) і аудит не є трьома окремими, послідовними дисциплінами, а трьома взаємопов'язаними функціями єдиного, безперервного циклу управління безпекою організації, що відповідає моделі трьох ліній захисту, розглянутій у підрозділі 17.3.

24.2. Модель зрілості SOC

Практична здатність організації реалізувати схему з підрозділу 24.1 не виникає одномоментно, а розвивається поетапно, що описується моделлю зрілості SOC. На **початковому рівні** організація має лише розрізнені, не інтегровані джерела подій і реагує на інциденти реактивно, без формалізованого процесу, часто дізнаючись про компрометацію лише через зовнішнє повідомлення. На **керованому рівні** впроваджено централізоване журналювання (розділ 2) і базовий SIEM (розділ 3), реагування частково формалізоване, але все ще значною мірою залежить від ручної роботи аналітика. На **визначеному рівні** SIEM доповнено Threat Intelligence (розділ 8), впроваджено формалізовані плейбуки реагування (розділ 10 модуля 2) і регулярний, планований аудит (розділ 19). На **кількісно керованому рівні** організація систематично вимірює ключові показники ефективності (підрозділ 24.6) і використовує UEBA/AI-аналіз (розділ 23) для пріоритизації. На **оптимізованому рівні**, найвищому, SOC-процес безперервно вдосконалюється на основі даних аудиту (розділи 17-22) і власної статистики інцидентів, включно з періодичним переглядом і оновленням моделей машинного навчання (підрозділ 23.4).

Практичний висновок: визначення поточного рівня зрілості власного SOC є першим, необхідним кроком перед плануванням будь-якого технологічного розширення, оскільки впровадження складних інструментів (наприклад, автономних AI-агентів, підрозділ 23.6) на організації, що ще не досягла керованого рівня, як правило, не приносить очікуваного ефекту.

24.3. Проектування SOC для гіпотетичної організації: вихідні умови

Для потреб комплексного практикуму цього розділу розглядається гіпотетична організація середнього розміру (умовно, регіональний банк чи великий обласний медичний заклад із власною ІТ-інфраструктурою): близько 800 робочих станцій, гібридна інфраструктура (частина серверів у власному дата-центрі, частина сервісів у хмарі відповідно до моделі спільної відповідальності з підрозділу 14.2

модуля 2), штат ІТ-відділу з двома виділеними фахівцями безпеки, регуляторні вимоги щодо захисту персональних даних клієнтів (підрозділ 7.8 модуля 1) і обмежений бюджет, що не дозволяє утримувати цілодобовий власний SOC.

Ці вихідні умови навмисно відображають типову, а не ідеалізовану ситуацію в українських організаціях середнього розміру, щоб практикум мав практичну, а не суто академічну цінність: рішення, прийняті в наступних підрозділах, послідовно враховують саме ці обмеження, а не виходять з необмежених ресурсів.

24.4. Вибір технологічного стеку та обґрунтування рішень

Виходячи з умов підрозділу 24.3, обґрунтованим вибором для журналювання й SIEM є рішення з відкритим кодом чи економічною ліцензійною моделлю (наприклад, стек, розглянутий у підрозділі 3.4 модуля 1, або Microsoft Sentinel, розділ 4, з огляду на модель оплати за обсяг даних, яка для організації середнього розміру може виявитися економічно виправданою порівняно з капітальними витратами на власну інфраструктуру Splunk, розділ 5). З огляду на обмежений штат фахівців безпеки (лише два), доцільним є розгляд часткового аутсорсингу функції цілодобового моніторингу до зовнішнього MSSP (підрозділ 1.10 модуля 1), із збереженням у штаті організації функцій другої й третьої ліній захисту (аналіз ескальованих інцидентів, адміністрування політик і незалежний внутрішній аудит, підрозділ 17.3), тоді як безперервне спостереження першої лінії передається постачальнику послуг.

Кожне таке рішення в реальному впровадженні повинне супроводжуватися формальним обґрунтуванням витрат і вигід (cost-benefit), що виходить за межі суто технічного розгляду цього підручника, але сама методологія вибору, зіставлення вимог організації (підрозділ 24.3) з можливостями конкретних інструментів, розглянутих у попередніх розділах, а не автоматичний вибір «найпотужнішого» рішення, є ключовим практичним навиком, який цей розділ покликаний закріпити.

24.5. Робочий процес обробки інциденту від виявлення до закриття

Повний робочий процес обробки інциденту для гіпотетичної організації з підрозділу 24.3 послідовно проходить такі етапи: **виявлення**, автоматичне спрацювання кореляційного правила SIEM чи оцінки ризику UEBA (розділ 23), або сповіщення від MSSP-партнера; **первинна тріаж-оцінка**, черговий аналітик першої лінії протягом узгодженого часу реагування (SLA) підтверджує чи відхиляє сигнал як хибнопозитивний; **ескалація**, підтверджений інцидент передається фахівцю

другої лінії організації разом із зібраним контекстом; **стримування й усунення**, застосування узгодженого плейбука реагування (підрозділ 10.5 модуля 2), включно, за потреби, з ізоляцією ураженого хоста; **відновлення**, повернення систем до нормального режиму роботи з підтвердженням FIM (розділ 21) про цілісність відновлених файлів; **пост-інцидентний розгляд (post-incident review)**, документування причин, часової шкали та засвоєних уроків; **закриття**, формальне завершення інциденту з передачею матеріалів для наступного планового аудиту (розділ 19).

Цей послідовний процес, хоча й описаний тут для єдиного гіпотетичного інциденту, на практиці масштабується на щоденний потік із десятків чи сотень сповіщень, що робить дисципліновану пріоритизацію (підрозділ 23.3) й чітко визначені ролі та SLA на кожному етапі не бажаною, а необхідною умовою керованості всього процесу.

24.6. Панель ключових показників ефективності SOC

Ефективність SOC-процесу, описаного в підрозділі 24.5, вимірюється набором формалізованих показників, аналогічно до дашбордів звітності, розглянутих у підрозділі 3.9 модуля 1. До ключових показників належать: **середній час виявлення (Mean Time to Detect, MTTD)**, інтервал між фактичним початком інциденту та моментом його виявлення; **середній час реагування (Mean Time to Respond, MTTR)**, інтервал між виявленням і початком стримування; **частка хибнопозитивних спрацювань**, показник, безпосередньо пов'язаний із точністю моделей із підрозділу 23.7; **відсоток покриття активів моніторингом**, частка систем організації, що дійсно надсилають події до SIEM, порівняно із загальною кількістю активів за інвентаризацією (підрозділ 9.3 модуля 2); і **кількість виявлених невідповідностей за плановий аудиторський цикл** (розділ 19), показник, що непрямо характеризує зрілість превентивних заходів контролю.

Жоден з цих показників не є самодостатнім: наприклад, штучне зниження MTTD шляхом послаблення порогів кореляційних правил, як правило, підвищує частку хибнопозитивних спрацювань, тому керівництво SOC повинне розглядати панель показників комплексно, а не оптимізувати кожен показник окремо у відриві від інших.

24.7. Перспективи розвитку курсу та подальшого вдосконалення SOC

Матеріал цього підручника охоплює фундаментальні принципи й основний сучасний технологічний інструментарій моніторингу, адміністрування та аудиту захищених ІТ-систем станом на момент його підготовки, однак технології SIEM, хмарної безпеки та застосування штучного інтелекту (розділ 23) розвиваються швидше, ніж цикл оновлення друкованого навчального видання. Тому засвоєння концептуальних принципів, моделі трьох ліній захисту (підрозділ 17.3), життєвого циклу аудиту (підрозділ 19.1), методології тріангуляції доказів (підрозділ 19.3) і формальних метрик оцінювання (підрозділи 23.7, 24.6), є для студента важливішим за запам'ятовування конкретних версій інтерфейсу конкретного продукту, оскільки саме ці принципи залишаються стабільними навіть за зміни конкретних технологічних рішень.

Для студентів, зацікавлених у поглибленому подальшому розвитку в цій галузі, доцільними напрямками є: практичне здобуття професійних сертифікацій, згаданих у відповідних розділах курсу (наприклад, за напрямом аудиту інформаційної безпеки чи конкретних хмарних платформ); участь у навчальних CTF-змаганнях (Capture The Flag) для практичного відпрацювання навичок виявлення й реагування; і самостійне поглиблене вивчення математичних основ машинного навчання для тих, хто прагне розвиватися саме в напрямі AI-аналізу подій безпеки (розділ 23), оскільки цей підручник свідомо викладає тему на концептуальному, а не суто математичному рівні, відповідному загальній програмі магістерської підготовки з кібербезпеки.

Освітню цінність CTF-формату для практичної підготовки фахівців з кібербезпеки детально проаналізовано в дослідженні Vasylenko, Grynkevych на прикладі проходження реальної навчальної машини платформи Hack The Box: автори демонструють, що симуляційний, змагальний формат дозволяє відпрацювати послідовність дій розвідки, експлуатації вразливості та підвищення привілеїв у контрольованому середовищі, наближеному до реального інциденту, не наражаючи продуктивну інфраструктуру організації на ризик, що робить CTF природним практичним доповненням до теоретичного матеріалу цього підручника.

24.8. Юридична та етична відповідальність SOC-аналітика

Аналітик SOC, розглянутий протягом усього цього розділу як центральна дійова особа процесу моніторингу й реагування, у своїй щоденній роботі отримує доступ до значного обсягу чутливої інформації, листування, особистих файлів,

поведінкових даних співробітників, що виявляється в ході розслідування інцидентів (підрозділ 24.4) чи роботи з DLP (розділ 16 модуля 2). Цей доступ покладає на аналітика особисту юридичну й етичну відповідальність, аналогічну до відповідальності, вже розглянутої стосовно впровадження DLP (підрозділ 16.8): інформація, отримана в ході розслідування, має використовуватися виключно в межах, необхідних для мети цього розслідування, а не для стороннього цікавості чи будь-якої мети, не пов'язаної з безпекою.

Формалізація цієї відповідальності на практиці набуває вигляду підписаної аналітиком угоди про нерозголошення (non-disclosure agreement) і кодексу етичної поведінки SOC, що визначає, зокрема, принцип «необхідно знати» (need to know) навіть усередині самої команди безпеки: доступ конкретного аналітика до конкретних даних розслідування обмежується колом осіб, справді залучених до цього розслідування, а не автоматично поширюється на всю команду. Порушення цієї довіри, наприклад несанкціоноване ознайомлення з особистими даними колеги поза межами офіційного розслідування, як правило, тягне для аналітика ті самі дисциплінарні й, за обставин, юридичні наслідки, що розглядалися в підрозділі 22.7 стосовно протидії аудиту, оскільки довіра до функції моніторингу й аудиту організації безпосередньо залежить від бездоганної особистої етики людей, які цю функцію виконують.

Практикум 24. Комплексне проєктування SOC-процесу для гіпотетичної організації

Мета роботи: самостійно спроектувати узгоджений SOC-процес, застосовуючи знання з усіх трьох модулів підручника одночасно.

Хід роботи:

1. на основі вихідних умов підрозділу 24.3 (або аналогічного варіанта, наданого викладачем) визначити поточний, найімовірніший рівень зрілості SOC гіпотетичної організації за моделлю з підрозділу 24.2 та обґрунтувати оцінку;
2. запропонувати й обґрунтувати вибір технологічного стеку моніторингу (SIEM/Sentinel/Splunk, розділи 3-5) з урахуванням бюджетних і кадрових обмежень організації;
3. описати повний робочий процес обробки типового інциденту (наприклад, підозра на компрометацію облікового запису адміністратора) за схемою підрозділу 24.5, з чіткими ролями та орієнтовними SLA на кожному етапі;
4. визначити, які саме заходи контролю з розділів 9-16 модуля 2 (адміністрування, IDS/IPS, мережева сегментація, хмара, Kubernetes) необхідні для стримування цього конкретного сценарію інциденту;
5. спроектувати план першого планового аудиту (розділи 17-19) новоствореного SOC-процесу, включно з переліком щонайменше п'яти конкретних об'єктів перевірки;
6. визначити щонайменше чотири ключові показники ефективності (підрозділ 24.6), якими керівництво організації оцінюватиме створений SOC-процес протягом першого року роботи;
7. підготувати короткий, структурований звіт (2-3 сторінки) з результатами виконання пунктів 1-6, що демонструє цілісне розуміння матеріалу курсу.

Контрольні питання до розділу 24

1. Поясніть, як розділи підручника, присвячені моніторингу, адмініструванню та аудиту, поєднуються в єдиний безперервний цикл управління безпекою.
2. Як архітектура з рисунка 24.1 співвідноситься з моделлю трьох ліній захисту з підрозділу 17.3?
3. Опишіть п'ять рівнів моделі зрілості SOC.
4. Чому впровадження складних AI-інструментів на організації низького рівня зрілості, як правило, не приносить очікуваного ефекту?

5. Які вихідні умови гіпотетичної організації підрозділу 24.3 є типовими для українських компаній середнього розміру?
6. Яку роль може відігравати частковий аутсорсинг до MSSP для організації з обмеженим штатом фахівців безпеки?
7. Чому вибір технологічного стеку повинен спиратися на зіставлення вимог організації з можливостями інструменту, а не на пошук «найпотужнішого» рішення?
8. Перелічіть усі етапи повного робочого процесу обробки інциденту від виявлення до закриття.
9. Що таке SLA в контексті обробки інциденту і на якому етапі процесу він застосовується?
10. Дайте означення MTTD і MTTR та поясніть різницю між ними.
11. Чому оптимізація одного показника ефективності SOC у відриві від інших може призвести до небажаних наслідків?
12. Яку роль відіграє плановий аудит (розділи 17-19) у циклі, зображеному на рисунку 24.1?
13. Чому концептуальні принципи курсу є важливішими для довгострокового засвоєння, ніж конкретні версії інтерфейсу продуктів?
14. Назвіть щонайменше три напрями подальшого професійного розвитку, запропоновані в підрозділі 24.7.
15. Поясніть власними словами, чому моніторинг, захист і аудит не можуть розглядатися як три незалежні одна від одної дисципліни.

ДОВІДКОВИЙ АПАРАТ

Глосарій термінів

Терміни Модуля 1 (моніторинг)

Моніторинг інформаційної безпеки — безперервний або регулярний процес спостереження, збору, реєстрації та первинного аналізу даних про стан захищеності ІТС (підрозділ 1.1).

ISCM (Information Security Continuous Monitoring) — визначене NIST SP 800-137 підтримання постійної обізнаності про інформаційну безпеку, вразливості

та загрози, достатньої для прийняття управлінських рішень щодо ризиків (підрозділ 1.1).

SNMP (Simple Network Management Protocol) — протокол активного моніторингу мережевого обладнання через опитування агентів і базу МІВ (підрозділ 1.5).

Базова лінія (baseline) — статистично усталений профіль нормальної поведінки системи чи суб'єкта, відносно якого оцінюється відхилення (підрозділ 1.7; поглиблено — підрозділ 23.2).

SIEM (Security Information and Event Management) — клас систем централізованого збору, кореляції та аналізу подій безпеки з різних джерел (розділ 3).

KQL (Kusto Query Language) — мова запитів, що використовується в Microsoft Sentinel для пошуку й аналізу подій (розділ 4).

SPL (Search Processing Language) — мова запитів платформи Splunk (розділ 5).

Threat Intelligence — розвідувальна інформація про загрози, зібрана, оброблена та інтегрована в SIEM для підвищення точності виявлення (розділ 8).

IoC (Indicator of Compromise) — технічна ознака (хеш файлу, IP-адреса, домен), що свідчить про ймовірну компрометацію (розділ 8).

DLP (Data Loss Prevention) — клас систем запобігання витоку конфіденційних даних через контрольовані канали передавання (розділ 7).

MSSP (Managed Security Service Provider) — зовнішній постачальник послуг моніторингу безпеки на аутсорсинговій основі (підрозділ 1.10).

Терміни Модуля 2 (адміністрування)

Адміністрування інформаційної системи — сукупність організаційних і технічних дій, спрямованих на підтримання працездатності, доступності та захищеності ІТС (підрозділ 9.1).

PDCA (Plan-Do-Check-Act) — цикл безперервного вдосконалення, застосований до адміністрування захищеної ІТС (підрозділ 9.3).

PoLP (Principle of Least Privilege) — принцип найменших привілеїв: користувач чи процес отримує лише мінімально необхідний для виконання задач рівень доступу (підрозділ 9.4).

CMDB (Configuration Management Database) — центральний репозиторій конфігураційних одиниць та зв'язків між ними (підрозділ 9.7).

IDS/IPS (Intrusion Detection/Prevention System) — системи виявлення та (для IPS) активного блокування мережових вторгнень (розділ 12).

RADIUS — протокол централізованої автентифікації, авторизації й обліку доступу користувачів до мережі (розділ 10).

VLAN (Virtual Local Area Network) — технологія логічного поділу фізичного комутатора на кілька незалежних ширококомовних доменів (підрозділ 13.8).

Zero Trust Network Access (ZTNA) — модель мережевого доступу, що не довіряє жодному суб'єкту за замовчуванням, незалежно від розташування в мережі (підрозділ 10.6).

PKI (Public Key Infrastructure) — інфраструктура відкритих ключів для видачі, перевірки та відкликання цифрових сертифікатів (підрозділ 10.9).

Спільна відповідальність (Shared Responsibility Model) — модель розподілу обов'язків із безпеки між хмарним провайдером і клієнтом (розділ 14).

RBAC (Role-Based Access Control) — керування доступом на основі ролей, застосовується зокрема в Kubernetes (розділ 15).

Service Mesh — інфраструктурний прошарок для керування взаємодією мікросервісів, включно з mTLS-шифруванням (підрозділ 15.7).

Infrastructure as Code (IaC) — підхід до опису й керування інфраструктурою через версійований програмний код, а не ручну конфігурацію (розділ 14; підрозділ 9.9).

Терміни Модуля 3 (аудит)

Аудит інформаційної безпеки — систематичний, незалежний і документований процес отримання доказів аудиту та їх об'єктивного оцінювання з метою визначення ступеня виконання узгоджених критеріїв аудиту (розділ 17).

Базова поведінка (baseline) — статистично усталений профіль типової активності суб'єкта (користувача, хоста), відносно якого система UEBA оцінює відхилення (розділ 23; вперше — підрозділ 8.7 модуля 1).

Дрейф моделі (model drift) — поступова втрата точності моделі машинного навчання внаслідок зміни реального середовища відносно даних, на яких модель була навчена (підрозділ 23.4).

Доказ аудиту — записи, викладення фактів або інша інформація, що стосується критеріїв аудиту і можуть бути перевірені (підрозділ 19.3).

Критерії аудиту — сукупність політик, процедур або вимог, які використовуються як еталон для порівняння з доказами аудиту (розділ 18).

Невідповідність (nonconformity) — недотримання вимоги, зафіксоване в ході аудиту (підрозділ 19.5).

Плейбук реагування — задокументована, покрокова послідовність дій для типового сценарію інциденту (підрозділ 10.5 модуля 2).

Протидія аудиту — сукупність технік приховування, спотворення чи знищення доказів, застосовуваних з метою уникнути виявлення порушення (розділ 22).

Три лінії захисту (Three Lines Model) — модель розподілу відповідальності за управління ризиками між операційним менеджментом, функціями ризик-менеджменту/комплаєнсу та незалежним внутрішнім аудитом (підрозділ 17.3).

UEBA (User and Entity Behavior Analytics) — клас систем поведінкової аналітики, що виявляють аномалії шляхом порівняння поточної активності суб'єкта з його історичним базовим профілем (підрозділ 8.7 модуля 1, розділ 23).

File Integrity Monitoring (FIM) — технологія контролю цілісності файлів шляхом порівняння поточного хешу з еталонним базовим значенням (розділ 21).

Timestomping — техніка навмисної зміни часових міток файлу з метою введення в оману аналітика (підрозділ 22.3).

Precision (точність) — частка сигналів, позначених моделлю як інцидент, що справді є істинними інцидентами (підрозділ 23.7).

Recall (повнота) — частка всіх реальних інцидентів, дійсно виявлених моделлю (підрозділ 23.7).

MTTD/MTTR (Mean Time to Detect/Respond) — середній час виявлення та середній час реагування на інцидент, ключові показники ефективності SOC (підрозділ 24.6).

SOC (Security Operations Center) — центр моніторингу безпеки, що об'єднує функції виявлення, аналізу та реагування на інциденти (розділ 24).

Перелік стандартів, згаданих у підручнику

Позначення	Повна назва
ISO 19011:2018	Guidelines for auditing management systems
ISO/IEC 27001:2022	Information security management systems. Requirements
ISO/IEC 27002:2022	Information security controls
NIST SP 800-53 Rev. 5	Security and Privacy Controls for Information Systems and Organizations
NIST SP 800-53A Rev. 5	Assessing Security and Privacy Controls
NIST SP 800-92	Guide to Computer Security Log Management
NIST SP 800-137	Information Security Continuous Monitoring (ISCM)
NIST SP 800-86	Guide to Integrating Forensic Techniques into Incident Response
NIST SP 800-147	BIOS Protection Guidelines
NIST Cybersecurity Framework 2.0	Framework for Improving Critical Infrastructure Cybersecurity
PCI DSS v4.0	Payment Card Industry Data Security Standard
ITU-T X.1205	Overview of cybersecurity

Словник аббревіатур

APT — Advanced Persistent Threat
CERT — Computer Emergency Response Team
CTF — Capture The Flag
DLP — Data Loss Prevention
EDR — Endpoint Detection and Response
FIM — File Integrity Monitoring
IDS/IPS — Intrusion Detection/Prevention System
KQL — Kusto Query Language
LLM — Large Language Model
MSSP — Managed Security Service Provider
MTTD/MTTR — Mean Time to Detect/Respond
NTP — Network Time Protocol
OCSP — Online Certificate Status Protocol
PKI — Public Key Infrastructure
RBAC — Role-Based Access Control
SIEM — Security Information and Event Management
SLA — Service Level Agreement
SOAR — Security Orchestration, Automation and Response
SOC — Security Operations Center
SPL — Search Processing Language
TLP — Traffic Light Protocol
UEBA — User and Entity Behavior Analytics
WORM — Write Once Read Many
XDR — Extended Detection and Response

Розширений перелік контрольних питань до іспиту

Наведений нижче повний перелік відтворює контрольні питання всіх 24 розділів курсу, згруповані за модулями, і призначений як єдиний інструмент фінального повторення перед заліком та іспитом, на додаток до глосарія й переліку стандартів вище.

Нижче наведено повний перелік контрольних питань усіх 24 розділів підручника, згрупований за модулями та розділами, для зручності підготовки до модульного контролю, заліку та іспиту. Питання відтворюють ті самі формулювання, що наведені в кінці кожного розділу.

Модуль 1. Моніторинг захищених ІТ-систем і мереж

Розділ 1. ОСНОВИ МОНІТОРИНГУ ЗАХИЩЕНИХ ІТ-СИСТЕМ І МЕРЕЖ

1. Чим моніторинг принципово відрізняється від аудиту та адміністрування?
2. Які рівні (об'єкти) моніторингу ІТС зображено на рисунку 1.2 і чому жоден з них не можна ігнорувати?
3. Що таке ISCM і яким документом NIST визначено це поняття?
4. Назвіть основні нормативні рівні регулювання моніторингу інформаційної безпеки в Україні.
5. Що таке MTTR і MTTR і чому саме ці показники обрані як головний критерій зрілості SOC?
6. Які операції передбачає протокол SNMP? Чим SNMPv3 відрізняється від SNMPv1/v2c і чому це критично для захищеної ІТС?
7. Що таке SOC і чим відрізняється власний SOC від MSSP?
8. Поясніть різницю між пасивним і активним моніторингом і наведіть приклад кожного.
9. Що таке alert fatigue і чому система, яка генерує занадто багато сповіщень, не виконує свого призначення?
10. Дайте означення показників False Positive Rate і False Negative Rate. Який з них зазвичай «дорожчий» для організації і чому?
11. Що таке EPS (Events Per Second) і навіщо цей параметр потрібен при плануванні SIEM-системи?
12. Порівняйте версії протоколу SNMP за таблицею 1.1: чому SNMPv1/v2c не рекомендовано використовувати в захищеній ІТС?

13. Які операції SNMP дозволяють пристрою самостійно, асинхронно повідомити систему керування про подію?
14. Назвіть три причини, чому організація впроваджує моніторинг ІТС.
15. На яких нормативних рівнях (від закону до міжнародного стандарту) регулюється моніторинг інформаційної безпеки в Україні?
16. Опишіть практичний сценарій, коли SOC-аналітик, який стежить лише за мережевим трафіком, пропустить реальний інцидент.
17. Які типи сканування безпеки ви можете назвати? Наведіть приклади інструментів для кожного типу.

Розділ 2. ЖУРНАЛЮВАННЯ ТА ЦЕНТРАЛІЗОВАНИЙ ЗБІР ПОДІЙ БЕЗПЕКИ

18. Чому журнали вважаються первинним доказовим матеріалом під час розслідування інцидентів?
19. Перелічіть основні джерела подій безпеки в типовій організації (таблиця 2.1).
20. Чим формат CEF відрізняється від «сирого» повідомлення syslog?
21. Які шість ланок утворюють конвеєр централізованого журналювання, зображений на рисунку 2.1?
22. Що таке нормалізація подій і чому без неї неможлива коректна кореляція? Наведіть приклади (CIM, ASIM).
23. Які вимоги висуваються до забезпечення цілісності журналів безпеки?
24. Назвіть щонайменше чотири типові джерела подій безпеки в організації.
25. Чим структурований формат журналювання (наприклад, JSON) відрізняється від неструктурованого текстового повідомлення?
26. Яку роль виконує агент або forwarder у конвеєрі централізованого збору подій (рисунок 2.1)?
27. Навіщо потрібен проміжний брокер повідомлень у деяких архітектурах збору журналів?
28. Що таке часова синхронізація (NTP) і чому вона критична для кореляції подій з різних джерел?
29. Наведіть приклад єдиної схеми нормалізації (CIM або ASIM) і поясніть, які поля вона визначає.
30. Скільки часу типово потрібно зберігати журнали безпеки і від чого залежить цей термін?
31. Що означає забезпечення цілісності журналів і якими технічними засобами воно досягається?

- 32. Чому «сирий» (raw) журнал варто зберігати навіть після нормалізації?
- 33. Опишіть різницю між журналюванням методом push (агент сам надсилає дані) і pull (система опитує джерело).

Розділ 3. SIEM-СИСТЕМИ: АРХІТЕКТУРА, ФУНКЦІЇ, ОГЛЯД РИНКУ РІШЕНЬ

- 34. Що означає абревіатура SIEM і з яких двох категорій засобів вона історично утворилася?
- 35. Опишіть чотири етапи еволюції технології, зображені на рисунку 3.1.
- 36. Перелічіть основні компоненти архітектури SIEM із рисунка 3.2 та поясніть роль кожного.
- 37. Порівняйте чотири методи кореляції подій із таблиці 3.1: який з них найкраще виявляє невідому раніше атаку і чому?
- 38. За якими критеріями обирають SIEM-рішення для організації?
- 39. Чому підручник свідомо не наводить рейтингу постачальників SIEM?
- 40. Яка різниця між SIM і SEM як історичними попередниками SIEM?
- 41. Перелічіть основні функціональні рівні архітектури SIEM.
- 42. Що таке правило кореляції (correlation rule) і наведіть приклад простого правила.
- 43. Чим кореляція на основі машинного навчання відрізняється від кореляції на основі фіксованих правил?
- 44. Що таке SOAR і як він пов'язаний із SIEM?
- 45. Які фактори варто врахувати при виборі SIEM-рішення для організації, крім вартості ліцензії?
- 46. Чим хмарний SIEM відрізняється від локально розгорнутого (on-premise)?
- 47. Що таке «час утримання» (retention) подій у SIEM і як він впливає на вартість рішення?
- 48. Наведіть приклад ситуації, коли надто «чутливе» правило кореляції шкодить роботі SOC.
- 49. Наведіть приклад, коли кореляція подій із кількох різних джерел виявляє атаку, непомітну за жодним джерелом окремо.

Розділ 4. MICROSOFT SENTINEL: ПОЗГОРТАННЯ ТА АДМІНІСТРУВАННЯ

- 50. На якому сховищі даних побудовано Microsoft Sentinel?
- 51. Перелічіть основні компоненти Sentinel із таблиці 4.1 та поясніть роль кожного.
- 52. Чим Fusion відрізняється від звичайного scheduled query rule?
- 53. Яку роль виконують Playbooks і на якій технології вони базуються?

54. Напишіть KQL-запит, що виводить кількість подій входу за останню годину, згруповану за користувачем.
55. Що таке Log Analytics Workspace і яку роль він відіграє в архітектурі Sentinel (рисунок 4.1)?
56. Чим Data Connector відрізняється від Analytics Rule?
57. Опишіть призначення Watchlists у Sentinel.
58. Що таке Fusion-правило і на чому базується його логіка виявлення?
59. Наведіть базову структуру KQL-запиту (from-where-project/summarize).
60. Чим NRT (Near Real-Time) правило відрізняється від звичайного scheduled query rule за частотою виконання?
61. Що таке Workbook у Sentinel і для чого він використовується?
62. Назвіть щонайменше три типи джерел даних, які можна підключити до Sentinel.
63. Чим ліцензування Sentinel (оплата за обсяг даних) відрізняється від традиційного ліцензування SIEM за кількістю вузлів?
64. Опишіть, як інцидент, створений Analytics Rule, потрапляє до черги реагування аналітика.
65. Яку перевагу дає інтеграція Sentinel з рештою продуктів екосистеми Microsoft (розділ 4.1)?

Розділ 5. SPLUNK: АРХІТЕКТУРА, ПОШУК ПОДІЙ, ДАШБОРДИ

66. Опишіть призначення трьох основних компонентів архітектури Splunk (Forwarder, Indexer, Search Head).
67. Чим index time відрізняється від search time?
68. Напишіть SPL-запит, що виводить кількість подій за годину для індексу vpn.
69. Що таке Risk-Based Alerting і яку проблему SOC він вирішує?
70. Наведіть приклад CIM-моделі та поясніть, навіщо вона потрібна.
71. Яку роль виконує Forwarder у архітектурі Splunk (рисунок 5.1) і чим Universal Forwarder відрізняється від Heavy Forwarder?
72. Що таке sourcetype і навіщо він потрібен під час індексування даних?
73. Опишіть базову структуру SPL-запиту з командами search, stats і table.
74. Чим індексований (indexed) час відрізняється від часу події (event time)?
75. Що таке Splunk Enterprise Security і яку функцію він додає до базового Splunk?
76. Навіщо потрібна Common Information Model (CIM) при побудові дашбордів і кореляційних пошуків?

- 77. Що таке Deployment Server і яку задачу він вирішує в розподіленій архітектурі Splunk?
- 78. Опишіть призначення Indexer Clustering і яку проблему воно вирішує.
- 79. Чим дашборд (dashboard) відрізняється від збереженого пошуку (saved search) у Splunk?
- 80. Наведіть приклад використання Risk-Based Alerting для зниження кількості хибнопозитивних інцидентів.
- 81. Порівняйте архітектурні підходи Splunk і Microsoft Sentinel: у чому головна відмінність моделі розгортання?

Розділ 6. ВИЯВЛЕННЯ МЕРЕЖЕВИХ АТАК ШЛЯХОМ АНАЛІЗУ ТРАФІКУ

- 82. Чим сигнатурний аналіз трафіку відрізняється від аналізу аномалій?
- 83. Чим PCAP-аналіз відрізняється від потокового (flow-based) аналізу і коли застосовується кожен з них?
- 84. Опишіть відмінність підходу Zeek від класичних сигнатурних IDS (Snort/Suricata).
- 85. Які ознаки в трафіку свідчать про DNS-tunneling (таблиця 6.1)?
- 86. Що таке «beaconing» і про що він може свідчити?
- 87. Порівняйте PCAP-аналіз і flow-based аналіз за обсягом даних, що зберігається, і глибиною деталізації (рисунок 6.1).
- 88. Що таке глибока інспекція пакетів (DPI) і чим вона відрізняється від аналізу заголовків пакета?
- 89. Наведіть приклад ознаки сканування портів (port scan) у мережевому трафіку.
- 90. Чим атака типу «людина посередині» (MITM) проявляється в аналізі трафіку?
- 91. Що таке NetFlow і які поля типово містить запис потоку (flow record)?
- 92. Чим IPFIX відрізняється від класичного NetFlow v5/v9?
- 93. Наведіть приклад того, як аномально велика кількість DNS-запитів до одного домену може свідчити про DNS-tunneling.
- 94. Чим Zeek (Bro) відрізняється концептуально від сигнатурних IDS на кшталт Snort?
- 95. Яку перевагу дає поєднання flow-based моніторингу з повними копіями пакетів (PCAP) для окремих сесій?
- 96. Опишіть практичний сценарій виявлення атаки перебору пароля (brute-force) за аналізом мережевого трафіку.
- 97. Чим система виявлення вторгнень мережевого рівня (розділ 6) відрізняється від повноцінного IDS/IPS-рішення, детально розглянутого в розділі 12 модуля 2?

Розділ 7. DLP-СИСТЕМИ: ОСНОВИ ЗАСТОСУВАННЯ

98. Чим мережева DLP відрізняється від кінцевої (endpoint) DLP?
99. Які методи детекції контенту застосовуються в сучасних DLP-системах (таблиця 7.1)?
100. Перелічіть основні компоненти архітектури SearchInform.
101. Які юридичні вимоги слід врахувати при впровадженні DLP-контролю дій співробітників?
102. Чому надто широкі політики DLP шкідливі для ефективності SOC?
103. Назвіть три типи DLP-систем за точкою розгортання (мережева, кінцева, для сховищ) і коротко опишіть кожен.
104. Що таке «цифровий відбиток» документа (digital fingerprint) і як він використовується для детекції?
105. Чим детекція за ключовими словами відрізняється від детекції на основі машинного навчання?
106. Перелічіть щонайменше чотири канали, які повинна контролювати повноцінна політика DLP.
107. Чим режим лише журналювання (monitor-only) відрізняється від режиму активного блокування в політиці DLP?
108. Опишіть архітектуру «агент-сервер», яку використовує SearchInform.
109. Чому надто широка політика DLP шкодить не лише зручності користувачів, а й ефективності роботи SOC?
110. Яким чином DLP-система може виявити спробу передавання лише фрагмента, а не всього конфіденційного документа?
111. Порівняйте задачі адміністрування DLP-системи і адміністрування IDS/IPS: що в них спільного щодо тюнінгу хибних спрацювань?
112. Наведіть приклад того, як DLP-контроль повинен бути узгоджений із внутрішньою політикою інформаційної безпеки організації.
113. Чому канал знімних носіїв (USB) залишається одним із найважче контрольованих для DLP-системи?

Розділ 8. THREAT INTELLIGENCE: ЗБІР, АНАЛІЗ, ІНТЕГРАЦІЯ З SIEM

114. Назвіть чотири рівні Threat Intelligence та поясніть різницю між ними.
115. Опишіть етапи життєвого циклу кіберрозвідки, зображені на рисунку 8.1.
116. Чим стандарт STIX відрізняється від TAXII?
117. Чим Indicator of Compromise відрізняється від Indicator of Attack?
118. Яку практичну роль виконує матриця MITRE ATT&CK у роботі SOC?

119. Як інтеграція Threat Intelligence із SIEM впливає на пріоритизацію інцидентів?
120. Опишіть різницю між стратегічним, оперативним, тактичним і технічним рівнями Threat Intelligence.
121. З яких шести етапів складається Intelligence Cycle, зображений на рисунку 8.1?
122. Чим комерційне джерело Threat Intelligence відрізняється від відкритого (OSINT)?
123. Що таке TIP (Threat Intelligence Platform) і яку функцію вона виконує?
124. Наведіть приклад того, як формат STIX структурує опис індикатора компрометації.
125. Яку роль виконує протокол TAXII у обміні даними Threat Intelligence між організаціями?
126. Чим тактика (tactic) відрізняється від техніки (technique) у матриці MITRE ATT&CK?
127. Що таке «збагачення» (enrichment) події за допомогою індикаторів компрометації?
128. Наведіть приклад ризику хибного індикатора компрометації (false IOC) і його впливу на роботу SOC.
129. Яким чином автоматизований аналіз телеметрії за допомогою II-агентів (розділ 1) пов'язаний із задачами Threat Intelligence?

Модуль 2. Адміністрування захищених ІТ-систем і мереж

Розділ 9. АДМІНІСТРУВАННЯ ІНФОРМАЦІЙНИХ СИСТЕМ І ПОЛІТИКА БЕЗПЕКИ ІТС

130. Чим адміністрування ІТС принципово відрізняється від моніторингу та аудиту?
131. Опишіть чотири рівні ієрархії документів політики інформаційної безпеки.
132. Яку мету переслідує кожен з чотирьох етапів циклу PDCA стосовно адміністрування ІТС?
133. У чому полягає принцип найменших привілеїв і чому підвищення прав повинно фіксуватися окремо?
134. Які ризики створює незадокументована зміна конфігурації продуктивної системи?
135. Наведіть приклад стандарту (другого рівня ієрархії з рисунка 9.1), що конкретизує вимогу політики безпеки.
136. Чому зміна стандарту не повинна вимагати перегляду всієї політики інформаційної безпеки?

137. Опишіть, що відбувається на кожному з чотирьох етапів циклу PDCA (рисунок 9.2) стосовно конкретної зміни конфігурації.
138. Дайте означення принципу найменших привілеїв (PoLP) і наведіть практичний приклад його порушення.
139. Чому системний адміністратор повинен мати окремий непривілейований обліковий запис для повсякденних задач?
140. Що таке підвищення прав (sudo, Run as Administrator, PIM) і чому воно має фіксуватися в журналі аудиту?
141. Перелічіть типові ролі адміністративного персоналу великої ІТС.
142. Що таке процес управління змінами (Change Management) і з яких етапів він складається?
143. Які ризики створює незадокументована зміна конфігурації продуктивної системи для безперервності бізнесу?
144. Чим підхід Infrastructure as Code (розділ 14) є логічним продовженням практики документування конфігурації?
145. Наведіть приклад робочої інструкції чи угоди SLA (четвертий рівень ієрархії політики) для конкретної системи.

*Розділ 10. ЗАХИСТ ІНФОРМАЦІЇ В КОМП'ЮТЕРНИХ МЕРЕЖАХ. ФУНКЦІЇ
АДМІНІСТРАТОРА БЕЗПЕКИ*

146. Чим обов'язки адміністратора безпеки відрізняються від обов'язків адміністратора системи чи мережі і чому бажане їх розділення?
147. Поясніть принцип Defense in Depth на прикладі мережі з рисунка 10.1.
148. У чому полягає різниця між автентифікацією, авторизацією та аудитом у моделі AAA?
149. Порівняйте протоколи RADIUS і TACACS+ за транспортом, шифруванням та типовим застосуванням.
150. Яку роль відіграє сегментація мережі в обмеженні наслідків компрометації одного вузла?
151. Перелічіть основні функції адміністратора безпеки в організації.
152. Чому бажане розділення обов'язків між адміністратором, що надає права доступу, і особою, що перевіряє коректність цього надання?
153. Опишіть, за яку частину загроз відповідає кожен з чотирьох рубежів на рисунку 10.1.
154. Наведіть приклад ситуації, коли інвестиції лише в периметровий міжмережевий екран створюють хибне відчуття захищеності.

- 155. Дайте означення кожного з трьох компонентів моделі AAA.
- 156. У яких сценаріях типово застосовують RADIUS, а в яких TACACS+ (таблиця 10.1)?
- 157. Що таке сегментація мережі і як вона обмежує наслідки компрометації одного вузла?
- 158. Чим міжмережевий екран класу NGFW відрізняється від традиційного пакетного фільтра?
- 159. Яку практику варто застосовувати адміністратору безпеки для підтримання правил міжмережевого екрана в актуальному стані?
- 160. Яку роль відіграє адміністратор безпеки під час реагування на інцидент?
- 161. Що таке формалізований план реагування на інциденти (Incident Response Plan) і чому його варто готувати заздалегідь?

Розділ 11. АДМІНІСТРУВАННЯ БАЗОВИХ ІНФОРМАЦІЙНИХ СЛУЖБ

- 162. Чому компрометація контролера домену вважається однією з найкритичніших подій для всієї ІТС?
- 163. Яку задачу вирішує DNSSEC і від якого класу атак він захищає?
- 164. Опишіть механізм атаки DHCP starvation і атаки rogue DHCP server. Чим вони відрізняються?
- 165. Яким технічним заходом на рівні комутатора протидіють несанкціонованому DHCP-серверу?
- 166. Що означають показники RPO і RTO і як вони впливають на вибір стратегії резервного копіювання?
- 167. Яку роль виконує контролер домену у взаємодії базових інформаційних служб (рисунок 11.1)?
- 168. Що таке tier-модель адміністрування контролерів домену і навіщо вона потрібна?
- 169. Чим DNS, інтегрована зі службою каталогів, відрізняється від автономного DNS-сервера?
- 170. Які записи (SPF, DKIM, DMARC) доповнюють DNSSEC для зниження ризику фішингових атак через підміну домену?
- 171. Опишіть механізм видачі адреси службою DHCP та основні параметри, що передаються клієнту.
- 172. Чим атака rogue DHCP server відрізняється від атаки DHCP starvation за метою злоумисника?

- 173. Що таке групові політики (Group Policy) і яку задачу централізованого управління вони вирішують?
- 174. Наведіть приклад аналога Group Policy в іншій екосистемі (Linux/хмара).
- 175. Що таке базовий профіль конфігурації (security baseline) і як централізоване управління конфігурацією його забезпечує?
- 176. Наведіть приклад, як цільові значення RPO і RTO впливають на вибір технології резервного копіювання.
- 177. Чому резервну копію необхідно періодично перевіряти на придатність до відновлення, а не лише створювати за розкладом?

*Розділ 12. СИСТЕМИ ВИЯВЛЕННЯ ВТОРГНЕНЬ (IDS/IPS):
АДМІНІСТРУВАННЯ*

- 178. Чим відрізняються NIDS і HIDS за точкою контролю та які загрози кожен з них здатен пропустити?
- 179. Поясніть різницю в мережевому підключенні між IDS (пасивний режим) та IPS (inline-режим) і наслідки цієї різниці для відмовостійкості мережі.
- 180. Порівняйте сигнатурний метод і метод виявлення аномалій за точністю та здатністю виявляти нові атаки.
- 181. Чим Suricata архітектурно відрізняється від Snort?
- 182. Якими засобами адміністратор може знизити частку хибнопозитивних спрацювань системи виявлення вторгнень, не втрачаючи здатності виявляти реальні атаки?
- 183. Яку задачу вирішує NIDS, яку не може вирішити HIDS, і навпаки?
- 184. Опишіть, чому відмова IPS може вплинути на роботу легітимного трафіку, а відмова IDS - ні (рисунок 12.1).
- 185. Що таке SPAN-порт (або мережевий TAP) і яку роль він відіграє для розгортання IDS?
- 186. У чому перевага сигнатурного методу виявлення атак і в чому його головне обмеження?
- 187. Чим атака нульового дня становить проблему саме для сигнатурного підходу?
- 188. Наведіть приклад того, як метод виявлення аномалій може дати хибнопозитивне спрацювання на легітимну активність.
- 189. Яку унікальну перевагу дає Zeek порівняно з класичними сигнатурними системами?
- 190. Що таке формат EVE JSON і чому він зручний для інтеграції Suricata з SIEM?

191. Перелічіть щонайменше три інструменти адміністратора для зниження частки хибнопозитивних спрацювань IDS/IPS.
192. З яких основних полів складається типове правило Snort/Suricata?
193. Наведіть приклад придушення (suppression) сповіщення для джерела, свідомо відомого як легітимне.

Розділ 13. ПОБУДОВА ЗАХИЩЕНИХ МЕРЕЖ ВІДПОВІДНО ДО МОДЕЛІ OSI

194. Чому недоцільно аналізувати захищеність мережі без розподілу на рівні моделі OSI?
195. Опишіть механізм атаки ARP spoofing і назвіть засіб протидії на канальному рівні.
196. У чому полягає фізичне розділення площини керування й площини передавання даних у концепції SDN?
197. Які нові ризики для безпеки мережі створює централізація керування в SDN-контролері?
198. Чим відрізняються завдання захисту на транспортному й на прикладному рівнях моделі OSI?
199. Чому недоцільно зосереджувати весь захист мережі лише на одному рівні моделі OSI?
200. Наведіть по одному прикладу типової загрози й засобу захисту для трьох довільних рівнів OSI (таблиця 13.1).
201. Опишіть механізм атаки VLAN hopping і засіб протидії на канальному рівні.
202. Чим статична прив'язка ARP відрізняється від динамічного протоколу ARP за рівнем захищеності?
203. Що таке автентифікація сусідів маршрутизації і від якої загрози вона захищає?
204. Який протокол найпоширеніше використовують для зв'язку між контролером SDN і мережевими пристроями?
205. Назвіть одну перевагу й один новий ризик, які централізація керування в SDN вносить для захищеності мережі.
206. Чим версія TLS 1.3 переважає TLS 1.2 з погляду рекомендацій NIST SP 800-52?
207. Що таке Web Application Firewall (WAF) і на якому рівні моделі OSI він переважно діє?
208. Чим практика Secure SDLC відрізняється від захисту вже розгорнутого застосунку лише засобами WAF?
209. Чому автентифікація і шифрування каналу керування є обов'язковою умовою безпечного впровадження SDN, а не опціональним доповненням?

*Розділ 14. ХМАРНА ІНФРАСТРУКТУРА: МОДЕЛІ, СПІЛЬНА
ВІДПОВІДАЛЬНІСТЬ, БЕЗПЕКА, АДМІНІСТРУВАННЯ*

- 210. Чим принципово відрізняються моделі послуг IaaS, PaaS і SaaS з погляду обсягу керування, що лишається на споживачеві?
- 211. Поясніть на власному прикладі, як модель спільної відповідальності визначає, хто відповідає за неправильно налаштоване хмарне сховище даних.
- 212. У чому полягає підхід Infrastructure as Code і чим він відрізняється від ручного налаштування через консоль керування?
- 213. Як опис інфраструктури у вигляді коду прискорює аварійне відновлення після втрати вузла чи сегмента?
- 214. Назвіть специфічні для хмарного середовища функції адміністрування, яких не існує (або які виражені значно слабше) в локальній інфраструктурі.
- 215. Дайте означення моделей IaaS, PaaS, SaaS і наведіть по одному прикладу типового продукту для кожної.
- 216. Чим приватна хмара відрізняється від гібридної за архітектурою розгортання?
- 217. Навіщо організації можуть свідомо обирати багатохмарну (multi-cloud) архітектуру?
- 218. Поясніть межу відповідальності за безпеку в моделі IaaS: що саме лишається на споживачеві (рисунок 14.1)?
- 219. Яка типова причина реальних інцидентів у хмарі, згідно з підрозділом 14.3?
- 220. Чим Terraform як інструмент оркестрації відрізняється від Ansible як системи управління конфігурацією?
- 221. Дайте означення показників RPO і RTO стосовно хмарної інфраструктури.
- 222. Чому наявність механізмів резервування в постачальника не звільняє організацію від власного плану аварійного відновлення?
- 223. Що таке тегування ресурсів (resource tagging) і яку практичну проблему воно вирішує в хмарному адмініструванні?
- 224. Чому надмірно широка IAM-політика вважається однією з найпоширеніших причин витоків даних із хмарних сховищ?
- 225. Які специфічні для хмари ризики пов'язані з помилково залишеним або скомпрометованим ресурсом і моделлю тарифікації за використання?
- 226. Чому регуляторні вимоги щодо розташування даних можуть переважити чисто економічну вигоду публічної хмари для державних систем?

*Розділ 15. КОНТЕЙНЕРИЗАЦІЯ ТА KUBERNETES: БЕЗПЕКА І
АДМІНІСТРУВАННЯ*

- 227. У чому головна архітектурна відмінність контейнера від віртуальної машини та які наслідки це має для швидкості розгортання?
- 228. Що таке багатоетапне складання образу Docker і чому воно корисне з погляду безпеки, а не лише розміру образу?
- 229. Перелічіть основні компоненти control plane Kubernetes і коротко опишіть роль кожного.
- 230. Як конвеєр CI/CD для контейнеризованого застосунку дозволяє виявляти проблеми безпеки до потрапляння коду в продуктивне середовище?
- 231. Поясніть, як RBAC і мережеві політики Kubernetes реалізують для кластера ті самі принципи, що розглядалися в розділах 9 і 10 для традиційної ІТС.
- 232. Чому контейнери запускаються значно швидше за віртуальні машини?
- 233. Що таке образ (image) і чим він відрізняється від запущеного контейнера?
- 234. Яку роль виконує API Server у control plane Kubernetes (рисунок 15.2)?
- 235. Для чого потрібен компонент etcd у кластері Kubernetes?
- 236. Чим Scheduler відрізняється від Controller Manager за функцією?
- 237. Що таке под (pod) і чому він, а не окремий контейнер, є найменшою одиницею розгортання в Kubernetes?
- 238. Опишіть типовий конвеєр CI/CD для контейнеризованого застосунку: з яких компонентів він складається?
- 239. Чому найдешевше й найефективніше інтегрувати перевірку безпеки саме на етапі CI, а не після розгортання застосунку?
- 240. Що таке RBAC у Kubernetes і який загальний принцип адміністрування безпеки він реалізує для кластера?
- 241. Яку задачу вирішують мережеві політики (Network Policies) у Kubernetes за замовчуванням і після налаштування?
- 242. Назвіть щонайменше три додаткові заходи безпеки контейнерів, окрім RBAC і мережевих політик.
- 243. Що таке сканування образів на вразливості і на якому етапі конвеєра його найдоцільніше виконувати?
- 244. Чому заборона запуску контейнера від імені root усередині образу вважається базовою практикою безпеки?

*Розділ 16. ПРАКТИКУМ: НАЛАШТУВАННЯ SEARCHINFORM ДЛЯ
КОНТРОЛЮ МЕРЕЖІ*

- 245. Яку архітектуру «агент-сервер» використовує SearchInform і за що відповідає кожен з цих компонентів?

- 246. Чим ретроспективний повнотекстовий пошук у зібраних даних відрізняється від сповіщення в реальному часі й для яких задач офіцера безпеки він потрібен?
- 247. Перелічіть типові канали передавання даних, які має контролювати політика DLP-системи.
- 248. Чому одноразового налаштування політики контролю під час впровадження, як правило, недостатньо для довгострокової ефективної роботи DLP-системи?
- 249. Порівняйте задачу тюнінгу хибнопозитивних спрацювань у DLP-системі із аналогічною задачею для IDS/IPS з розділу 12: що в них спільного?
- 250. Опишіть роль агента-перехоплювача на робочій станції в архітектурі SearchInform (рисунок 16.1).
- 251. Яку функцію виконує централізований повнотекстовий індекс сервера SearchInform?
- 252. Що визначає політика контролю мережевого каналу? Перелічіть щонайменше три складові.
- 253. Чим виявлення чутливого вмісту за цифровими відбитками відрізняється від виявлення за ключовими словами?
- 254. Чим відрізняється дія системи «лише журналювання» від дії «активне блокування»?
- 255. Чому надто широка політика контролю створює потік хибнопозитивних спрацювань, аналогічний проблемі IDS/IPS?
- 256. Які завдання адміністрування покладаються на офіцера інформаційної безпеки, що працює з SearchInform?
- 257. Наведіть приклад граничного випадку, який ускладнює налаштування політики контролю (наприклад, передавання невеликого фрагмента документа).
- 258. Чому одноразового налаштування політики DLP під час впровадження зазвичай недостатньо для довгострокової ефективності?
- 259. Які канали передавання даних, окрім електронної пошти, повинна контролювати повноцінна політика DLP?
- 260. Чим ретроспективний аналіз перехопленого трафіку доповнює можливості SIEM-системи з розділу 3?

Модуль 3. Аудит захищених ІТ-систем і мереж

Розділ 17. ОСНОВНІ ПІДХОДИ ДО МОНІТОРИНГУ ТА АУДИТУ БЕЗПЕКИ ІНФОРМАЦІЇ

- 261. Дайте означення аудиту інформаційної безпеки і поясніть значення слів «систематичний», «незалежний», «документований» у цьому визначенні.
- 262. Чим внутрішній аудит принципово відрізняється від зовнішнього?
- 263. Що таке сертифікаційний аудит і чим він відрізняється від звичайного зовнішнього аудиту?
- 264. Опишіть модель трьох ліній захисту (рисунок 17.1) і роль кожної лінії.
- 265. До якої лінії моделі належить SOC-аналітик? А внутрішній аудитор?
- 266. Які типові об'єкти аудиту в захищеній ІТС?
- 267. Що таке критерії аудиту (audit criteria) і чому аудит неможливий без їх заздалегідь чіткого визначення?
- 268. Перелічіть сім принципів аудиту за ISO 19011.
- 269. У чому полягає ризик-орієнтований підхід до аудиту на відміну від рівномірної «килимової» перевірки?
- 270. Опишіть ролі замовника аудиту, керівника групи аудиту, аудитора й аудованого суб'єкта.
- 271. Яку роль в аудиті виконує технічний експерт і чому він не є аудитором?
- 272. Чим аудит відрізняється від комплаєнсу?
- 273. Чим аудит відрізняється від тестування на проникнення за метою й методом?
- 274. Чому зріла організація поєднує аудит, комплаєнс і тестування на проникнення, а не обирає щось одне?
- 275. Що таке програма аудиту організації і на який горизонт планування вона зазвичай розраховується?
- 276. Чому програма аудиту рідко здатна охопити абсолютно всі об'єкти організації щороку?
- 277. Наведіть приклад того, як принцип незалежності аудиту порушується, якщо адміністратор перевіряє власні налаштування.

Розділ 18. СТАНДАРТИ ТА МЕТОДИКИ АУДИТУ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ

- 278. Який статус має стандарт ISO 19011: чи сертифікується відповідність йому і чому?
- 279. Опишіть етапи сертифікаційного циклу ISO/IEC 27001: Stage 1, Stage 2, наглядові аудити, ресертифікація.
- 280. Чим стандарт ISO/IEC 27001 відрізняється від ISO/IEC 27002?
- 281. Що таке NIST SP 800-53A і як він пов'язаний з каталогом NIST SP 800-53?

- 282. Назвіть рівні української нормативної бази, релевантні для аудиту інформаційної безпеки.
- 283. Що таке НД ТЗІ і для яких організацій їх виконання обов'язкове?
- 284. Для яких організацій обов'язковий аудит PCI DSS?
- 285. Чим звіт SOC 2 типу II відрізняється від одноразової перевірки на конкретну дату?
- 286. Порівняйте чекліст-орієнтований і ризик-орієнтований підходи до аудиту.
- 287. Чи є чекліст протилежністю ризик-орієнтованого підходу? Обґрунтуйте відповідь.
- 288. Що таке матриця відповідності (compliance mapping) і яку практичну проблему вона вирішує?
- 289. Наведіть приклад заходу контролю, що одночасно покриває вимоги двох різних стандартів.
- 290. Якими факторами визначається вибір методики аудиту для конкретної організації?
- 291. Чому вибір стандарту аудиту рідко є суто технічним рішенням?
- 292. Порівняйте підхід NIST (каталог 800-53/800-53A) і підхід ISO/IEC 27001 за рівнем деталізації вимог.
- 293. Що таке QSA і яку роль ця роль відіграє в аудиті PCI DSS?

*Розділ 19. ЗАХОДИ ЩОДО АУДИТУ ЗАХИЩЕНИХ ІТ-СИСТЕМ: ЕТАПИ,
ЗБІР ДОКАЗІВ, ЗВІТНІСТЬ*

- 294. Перелічіть п'ять етапів життєвого циклу аудиторської перевірки за ISO 19011.
- 295. Що визначає план аудиту і хто його затверджує?
- 296. Що таке вибірка (sampling) в аудиті і чому вона часто необхідна?
- 297. Перелічіть чотири основні методи збору доказів аудиту.
- 298. Що таке тріангуляція доказів і чому вона підвищує переконливість висновку?
- 299. Що таке робочі документи аудитора і яку функцію вони виконують?
- 300. Порівняйте функцію робочих документів аудитора й ланцюга зберігання доказів (підрозділ 2.10 модуля 1).
- 301. Опишіть чотири рівні критичності невідповідностей за результатами аудиту.
- 302. Чому невідповідність повинна бути класифікована, а не просто зафіксована як факт?
- 303. З яких основних розділів складається звіт аудиту?

- 304. Чому звіт аудиту повинен містити не лише перелік проблем, а й позитивні висновки?
- 305. Що таке план коригувальних дій (CAP) і хто його розробляє?
- 306. Чому саме аудований суб'єкт, а не аудитор, розробляє план коригувальних дій?
- 307. Що таке подальший контроль (follow-up) і навіщо він потрібен?
- 308. Що таке забезпечення якості аудиту і які два його основні механізми?
- 309. Наведіть приклад помилки на етапі планування аудиту, яка здатна підірвати достовірність усього звіту.

*Розділ 20. АУДИТ З ВИКОРИСТАННЯМ СКАНЕРІВ БЕЗПЕКИ
(ПОГЛИБЛЕНО)*

- 310. Чим сканування як метод збору доказів аудиту відрізняється від того самого сканування, виконаного адміністратором для власних потреб?
- 311. Перелічіть три класи сканерів безпеки та відмінність у запитаннях, на які кожен з них відповідає.
- 312. Що таке Rules of Engagement і чому вони обов'язкові перед скануванням у межах аудиту?
- 313. Яка юридична різниця між авторизованим і неавторизованим скануванням тієї самої мережі?
- 314. Що таке дедуплікація результатів сканування?
- 315. Що таке верифікація знайденої вразливості і чому вона важлива для якості доказу аудиту?
- 316. Порівняйте сканування вразливостей і тестування на проникнення за глибиною й охопленням.
- 317. На яке запитання відповідає сканування вразливостей, а на яке, тестування на проникнення?
- 318. Чому одноразового сканування під час аудиторської перевірки недостатньо для повноцінного управління вразливостями?
- 319. Що таке комплаєнс-сканування конфігурації і чим воно відрізняється від сканування вразливостей?
- 320. Наведіть приклад еталонного профілю для комплаєнс-сканування.
- 321. Яку кількісну оцінку здатне надати комплаєнс-сканування для звіту аудиту?
- 322. Назвіть щонайменше два типи проблем, які автоматизоване сканування принципово не здатне виявити.
- 323. Чому кінцева відповідальність за висновок аудиту завжди лишається за аудитором-людиною?

324. Як пов'язані регулярне автоматизоване сканування (перша/друга лінія) і періодичний аудит (третя лінія) з погляду моделі трьох ліній?

Розділ 21. АУДИТ ЦІЛІСНОСТІ ФАЙЛОВИХ СИСТЕМ

325. Дайте означення цілісності даних і поясніть, чому це одна з трьох класичних властивостей інформаційної безпеки.

326. Опишіть принцип роботи FIM на основі хеш-функцій (рисунок 21.1).

327. Чому для FIM не рекомендовано застосовувати алгоритми MD5 чи SHA-1?

328. Назвіть відомі інструменти FIM, комерційні й з відкритим кодом.

329. Чому базовий профіль повинен будуватися лише на достовірно довірєній системі?

330. Які ризики пов'язані з надто широким переліком контрольованих файлів у FIM? А з надто вузьким?

331. Який стандарт прямо вимагає впровадження FIM і для яких систем?

332. Чому наявність чи відсутність FIM є зручним, легко верифіковним доказом аудиту?

333. Що спричиняє хибнопозитивні спрацювання FIM?

334. Як інтеграція FIM з процесом управління змінами (розділ 9 модуля 2) знижує кількість хибнопозитивних спрацювань?

335. Чому класична модель FIM погано підходить для контейнеризованого середовища?

336. Що таке підписування образів (image signing) і як воно замінює класичний FIM у Kubernetes?

337. Порівняйте превентивний (image signing) і детективний (класична FIM) підходи до цілісності в контексті ешелонованого захисту.

338. Наведіть приклад санкціонованої й несанкціонованої зміни файлової системи.

339. Чим FIM концептуально схожа на цифровий відбиток документа з DLP (підрозділ 7.2 модуля 1)?

*Розділ 22. ДОСЛІДЖЕННЯ ЗАСОБІВ ПЕРЕШКОДЖАННЯ АУДИТУ
ІНФОРМАЦІЇ В ІТ-СИСТЕМІ*

340. Дайте означення протидії аудиту та поясніть дві категорії мотивації її застосування.

341. Чим очищення журналу відрізняється від вибіркового редагування журналу?

342. Чому спроба очищення журналу подій Windows сама по собі часто залишає слід?

- 343. Що таке стеганографія і як вона застосовується для приховування даних у файловій системі?
- 344. Що таке альтернативні потоки даних (ADS) і в якій файловій системі ця можливість присутня?
- 345. Що таке timestomping і яку саме властивість доказів він атакує?
- 346. Чим протидія скануванню в межах аудиту (підрозділ 22.4) технічно споріднена з ухиленням від IDS/IPS (підрозділ 12.6 модуля 2)?
- 347. Чому конфіденційність точного графіка сканування має значення для ефективності аудиту?
- 348. Перелічіть щонайменше чотири заходи захисту, що ускладнюють протидію аудиту, і поясніть механізм дії кожного.
- 349. Чому FIM лишається ефективним індикатором навіть за застосування timestomping?
- 350. Назвіть щонайменше три «червоні прапорці», що можуть свідчити про протидію аудиту.
- 351. Чому невідповідність між локальною й централізованою копією журналу є вагомим підставою для поглибленого розслідування?
- 352. Чи кваліфікують знайдені ознаки протидії аудиту як звичайну невідповідність? Обґрунтуйте відповідь.
- 353. Які юридичні наслідки може мати навмисне знищення доказів у процесі офіційного розслідування?
- 354. Чому знищення журналу як таке часто тягне окремі юридичні наслідки, незалежно від наслідків первинного порушення, яке воно намагалося приховати?

Розділ 23. AI-АНАЛІЗ ПОДІЙ БЕЗПЕКИ: МАШИННЕ НАВЧАННЯ, UEBA В SIEM/SOC

- 355. Назвіть три типи задач машинного навчання та наведіть приклад застосування кожного до аналізу подій безпеки.
- 356. Чому навчання без учителя є базовим підходом для UEBA?
- 357. Що таке базовий профіль поведінки суб'єкта і які параметри він охоплює?
- 358. Чому початковий період навчання моделі UEBA є вразливим вікном з погляду безпеки?
- 359. Опишіть послідовність етапів типового конвеєра обробки подій засобами машинного навчання.

- 360. Яку роль відіграє контур зворотного зв'язку аналітика в підвищенні точності моделі?
- 361. Що таке дрейф моделі і чому він вимагає регулярного перенавчання?
- 362. Що таке змагальне машинне навчання і чим воно споріднене з протидією аудиту з розділу 22?
- 363. Що таке надмірна довіра до автоматизованого результату (automation bias) і чому вона небезпечна саме в контексті аудиту?
- 364. Для яких задач у SOC застосовуються генеративні мовні моделі (LLM)?
- 365. Чому результати, згенеровані LLM, потребують обов'язкової верифікації аналітиком?
- 366. Яку роль відіграють автономні AI-агенти для виявлення мережових інцидентів і помилок конфігурації, за прикладом дослідження Rudin, Vasylenko, Grynkevych (2025)?
- 367. Чому доцільно впроваджувати автономність AI-агента поступово, а не одразу повністю?
- 368. Дайте означення точності (precision) та повноти (recall) і поясніть різницю між ними.
- 369. Чому низька повнота є найнебезпечнішим типом помилки саме для задач безпеки?
- 370. Які питання приватності виникають при впровадженні UEBA?
- 371. Що таке алгоритмічна упередженість моделі і як її можна виявити?

*Розділ 24. КОМПЛЕКСНИЙ ПРАКТИКУМ: ПОБУДОВА SOC-ПРОЦЕСУ
МОНІТОРИНГУ, АУДИТУ ТА РЕАГУВАННЯ*

- 372. Поясніть, як розділи підручника, присвячені моніторингу, адмініструванню та аудиту, поєднуються в єдиний безперервний цикл управління безпекою.
- 373. Як архітектура з рисунка 24.1 співвідноситься з моделлю трьох ліній захисту з підрозділу 17.3?
- 374. Опишіть п'ять рівнів моделі зрілості SOC.
- 375. Чому впровадження складних AI-інструментів на організації низького рівня зрілості, як правило, не приносить очікуваного ефекту?
- 376. Які вихідні умови гіпотетичної організації підрозділу 24.3 є типовими для українських компаній середнього розміру?
- 377. Яку роль може відігравати частковий аутсорсинг до MSSP для організації з обмеженим штатом фахівців безпеки?

- 378. Чому вибір технологічного стеку повинен спиратися на зіставлення вимог організації з можливостями інструменту, а не на пошук «найпотужнішого» рішення?
- 379. Перелічіть усі етапи повного робочого процесу обробки інциденту від виявлення до закриття.
- 380. Що таке SLA в контексті обробки інциденту і на якому етапі процесу він застосовується?
- 381. Дайте означення MTTD і MTTR та поясніть різницю між ними.
- 382. Чому оптимізація одного показника ефективності SOC у відриві від інших може призвести до небажаних наслідків?
- 383. Яку роль відіграє плановий аудит (розділи 17-19) у циклі, зображеному на рисунку 24.1?
- 384. Чому концептуальні принципи курсу є важливішими для довгострокового засвоєння, ніж конкретні версії інтерфейсу продуктів?
- 385. Назвіть щонайменше три напрями подальшого професійного розвитку, запропоновані в підрозділі 24.7.
- 386. Поясніть власними словами, чому моніторинг, захист і аудит не можуть розглядатися як три незалежні одна від одної дисципліни.

Додаток. Приклади типових конфігурацій та запитів

Наведені нижче фрагменти є ілюстративними навчальними прикладами типового синтаксису відповідних мов і форматів, згаданих у розділах підручника. Перед практичним використанням синтаксис слід звіряти з чинною документацією конкретної версії продукту (підрозділ 4.1, примітка щодо цього вже наводилася в основному тексті).

Приклад 1. Запит KQL (Microsoft Sentinel) — пошук облікових записів з аномальною кількістю невдалих спроб входу за годину (розділ 4)

```
SigninLogs
| where TimeGenerated > ago(1h)
| where ResultType != "0"
| summarize FailedCount = count() by UserPrincipalName, bin(TimeGenerated, 1h)
| where FailedCount > 10
| order by FailedCount desc
```

Приклад 2. Запит SPL (Splunk) — побудова дашборду автентифікацій (розділ 5)

```
index=auth sourcetype=linux_secure
| stats count by user, action
| where action="failure"
| sort - count
```

Приклад 3. Фрагмент правила Suricata для виявлення сканування портів (розділ 12)

```
alert tcp any any -> $HOME_NET any (msg:"Можливе сканування портів";
flags:S; threshold:type both, track by_src, count 20, seconds 10;
sid:1000010; rev:1;)
```

Приклад 4. Kubernetes NetworkPolicy — обмеження мережевого трафіку до простору імен (розділ 15)

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: deny-external-traffic
  namespace: production
spec:
  podSelector: {}
  policyTypes:
  - Ingress
  ingress:
```

```

-
-
from:
namespaceSelector:
matchLabels:
team: backend

```

Приклад 5. Фрагмент чек-листа планового аудиту за принципами ISO 19011 (розділ 19)

№	Об'єкт перевірки	Критерій	Метод збору доказів	Результат
1	Політика паролів	Мінімум 12 символів, MFA для адміністраторів	Перегляд конфігурації AD/LDAP	відповідність / невідповідність
2	Журнали подій	Централізоване зберігання, WORM, ≥ 12 міс.	Перевірка конфігурації SIEM-конектора	відповідність / невідповідність
3	Патч-менеджмент	Критичні патчі — протягом 72 год.	Вибірка систем, звіт сканера вразливостей	відповідність / невідповідність
4	Резервне копіювання	Щоденне, тестове відновлення щоквартально	Журнал завдань бекапу, протокол тесту відновлення	відповідність / невідповідність
5	Права доступу	Перегляд прав щоквартально, PoLP	Вибірка облікових записів, порівняння з посадовими обов'язками	відповідність / невідповідність

Приклад 6. Фрагмент SNMP-запиту через утиліту командного рядка (розділ 1)

```
snmpget -v2c -c public 192.0.2.10 1.3.6.1.2.1.1.3.0
```

Примітка: усі наведені приклади спрощено ілюструють синтаксис і призначені виключно для навчальних цілей; перед використанням у продуктивному

середовищі кожен приклад потребує адаптації до конкретної інфраструктури організації.

Зведений список використаних джерел

Цей розділ об'єднує в одному місці всі джерела, використані в розділах усіх трьох модулів підручника (наукова та навчально-методична література, стандарти, нормативно-правові акти), без повторів у межах усього видання. Повний перелік джерел кожного окремого розділу наведено також безпосередньо після цього розділу в тексті відповідного розділу.

1. Бурячок В.Л., Гулак Г.М., Толубко В.Б. Інформаційний та кіберпростори: проблеми безпеки, методи та засоби боротьби: підручник. Київ: ТОВ «СІК ГРУП УКРАЇНА», 2015. 449 с.
2. Бурячок В.Л., Толюпа С.В., Аносов А.О., Козачок В.А., Лукова-Чуйко Н.В. Системний аналіз та прийняття рішень в інформаційній безпеці: підручник. Київ: ДУТ, 2015. 345 с.
3. Vasylenko V., Grynkevych G., Kuznietsov I. The Role of Capture-the-Flag (CTF) Challenges in Cybersecurity Research and Training: Analysis of the «Editorial» Machine. Кібербезпека: освіта, наука, техніка. 2025. Т. 4, № 28. С. 137-149. DOI: 10.28925/2663-4023.2025.28.762.
4. Гринкевич Г.О. Аспекти безпеки та конфігурація несправностей при розгортанні SDN мереж. Сучасний захист інформації. 2016. № 3. С. 3–8.
5. Гринкевич Г.О., Василенко В.В., Ковлєва Т.М., Москаленко А.О., Легомінов В.О. Аналіз програмного забезпечення, необхідного для контролю за пакетами даних у програмно-конфігурованих мережах. Наукові записки Українського науково-дослідного інституту зв'язку. 2019. № 1(53). С. 42–49.
6. Grynkevych G., Vasylenko V., Rudin D. Local Open-Source Platform for Cyber Incident Monitoring and Response with AI Support as an Alternative to EDR/XDR Solutions. Кібербезпека: освіта, наука, техніка. 2026. Т. 1, № 33. С. 234–241. DOI: 10.28925/2663-4023.2026.33.1211.
7. Гринкевич Г.О., Домрачева К.О., Шелудько С.В., Коновалов Д.П. Підходи до вимірювання та моніторингу мережі для забезпечення перевірки керуючих повідомлень OpenFlow. Зв'язок. 2018. № 5. С. 30–35.
8. Гринкевич Г.О., Макаренко А.О., Миронов Д.В., Жебка В.В. Моніторинг телекомунікаційних мереж: навчальний посібник. Київ: ДУТ, 2018. 351 с.
9. Гринкевич Г. О., Макаренко А. О., Жебка В. В., Куклов В. М., Підручний А. І. Аналіз працездатності програмно-конфігурованих мереж. Наукові записки Українського науково-дослідного інституту зв'язку. 2017. № 1(45). С. 32-39.

10. Документація програмного комплексу SearchInform (чинна версія, за навчальним доступом кафедри).
11. ДСТУ ISO/IEC 27001:2023. Інформаційна безпека, кібербезпека та захист приватності. Системи управління інформаційною безпекою. Вимоги.
12. ДСТУ 3396.2-97. Захист інформації. Технічний захист інформації. Терміни та визначення.
13. Закон України «Про захист персональних даних» від 01.06.2010 № 2297-VI.
14. Закон України «Про основні засади забезпечення кібербезпеки України» від 05.10.2017 № 2163-VIII.
15. Інформаційні технології. Звід правил з управління захистом інформації: ISO/IEC 27002:2005 (E). Харків: Компанія «Технормінатив», 2007. 117 с.
16. Комп'ютерні мережі: навч. посіб. для технічних спец. вищих навч. закл. Кн. 2. Львів: Магнолія 2006, 2014. 327 с.
17. Кононович В.Г., Гладиш С.В. Технічна експлуатація систем захисту інформації телекомунікаційних мереж загального користування. Ч. 4. Інформаційна безпека комунікаційних мереж та послуг. Реагування на атаки: навчальний посібник. Одеса: ОНАЗ ім. О.С. Попова, 2009. 208 с.
18. Курило А.П. Аудит інформаційної безпеки. Київ: БДЦ-прес, 2006. 304 с.
19. Lavriv O., Buhyl B., Klymash M., Grynkevych G. Services Continuous Integration Based on Modern Free Infrastructure. 2017 2nd International Conference on Advanced Information and Communication Technologies (AICT). Lviv, 2017. P. 150-154. DOI: 10.1109/AIACT.2017.8020087.
20. Lavriv O., Klymash M., Grynkevych G., Tkachenko O., Vasylenko V. Method of Cloud System Disaster Recovery Based on «Infrastructure as a Code» Concept. 2018 14th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET). Lviv-Slavske, 2018. DOI: 10.1109/TCSET.2018.8336395.
21. Rudin D., Vasylenko V., Grynkevych G. Adaptive Network Security Automation: DevSecOps, ML Detection, and Policy-as-Code. International Scientific Journal «Internauka». 2026. № 1. DOI: 10.25313/2520-2057-2026-1-12020.
22. Rudin D., Vasylenko V., Grynkevych G. AI Agents for Automated Network Incident Detection and Misconfiguration Identification. International Scientific Journal «Internauka». 2025. № 12. DOI: 10.25313/2520-2057-2025-12-12019
23. Rudin D., Vasylenko V., Grynkevych G. Data Protection in Wi-Fi Networks of Financial Institutions: WEP to WPA3 Evolution and Economic Impact. International

- Scientific Journal «Internauka». 2025. № 8. DOI: 10.25313/2520-2057-2025-8-12018.
24. Толюпа С.В., Хращевський Р.В., Гринкевич Г.О., Макаренко А.О., В'юнник О.В. Управління та якість послуг інформаційних мереж зв'язку: навчальний посібник. Київ: Логас; Державний університет телекомунікацій, 2014. 361 с. Гриф Міністерства освіти і науки України № 1/11-6442 від 30.04.2014.
 25. Center for Internet Security. CIS Benchmarks and CIS Controls.
 26. Center for Internet Security. CIS Docker Benchmark; CIS Kubernetes Benchmark.
 27. Cisco Systems. TACACS+ Protocol, Version 1.78. Technical Specification.
 28. Cloud Security Alliance. Security Guidance for Critical Areas of Focus in Cloud Computing, v4.
 29. Defense Information Systems Agency. Security Technical Implementation Guides (STIG).
 30. IEEE 802.1X-2020. IEEE Standard for Local and Metropolitan Area Networks: Port-Based Network Access Control.
 31. ISO 19011:2018. Guidelines for auditing management systems.
 32. ISO/IEC 27001:2022. Information security, cybersecurity and privacy protection. Information security management systems. Requirements.
 33. ISO/IEC 27002:2022. Information security, cybersecurity and privacy protection. Information security controls.
 34. ITIL 4 Foundation: IT Service Management. Change Enablement Practice.
 35. Kubernetes Documentation. Concepts: Overview of Kubernetes, Cluster Architecture, RBAC, Network Policies.
 36. Microsoft Learn. Документація Microsoft Sentinel. Microsoft Corporation. URL: <https://learn.microsoft.com/azure/sentinel/>
 37. Microsoft Learn. Active Directory Domain Services Documentation: Tiered Administration Model.
 38. Microsoft Learn. Kusto Query Language (KQL) overview. Microsoft Corporation. URL: <https://learn.microsoft.com/kusto/query/>
 39. MISP Project. MISP — Open Source Threat Intelligence Platform documentation. URL: <https://www.misp-project.org>
 40. MITRE ATT&CK. Матриця тактик і технік для корпоративного середовища. MITRE Corporation. URL: <https://attack.mitre.org>
 41. MITRE ATT&CK. Adversarial Tactics, Techniques, and Common Knowledge Framework.

42. MITRE ATT&CK. MITRE Corporation. URL: <https://attack.mitre.org>
43. MITRE ATT&CK. Tactic: Defense Evasion. MITRE Corporation. URL: <https://attack.mitre.org/tactics/TA0005/>
44. NIST Cybersecurity Framework 2.0. National Institute of Standards and Technology, 2024.
45. NIST Special Publication 800-115. Technical Guide to Information Security Testing and Assessment. National Institute of Standards and Technology, 2008.
46. NIST Special Publication 800-137. Information Security Continuous Monitoring (ISCM) for Federal Information Systems and Organizations. National Institute of Standards and Technology, 2011.
47. NIST Special Publication 800-144. Guidelines on Security and Privacy in Public Cloud Computing.
48. NIST Special Publication 800-145. The NIST Definition of Cloud Computing.
49. NIST Special Publication 800-147. BIOS Protection Guidelines. National Institute of Standards and Technology, 2011.
50. NIST Special Publication 800-41 Rev. 1. Guidelines on Firewalls and Firewall Policy.
51. NIST Special Publication 800-52 Rev. 2. Guidelines for the Selection, Configuration, and Use of TLS Implementations.
52. NIST Special Publication 800-53A Rev. 5. Assessing Security and Privacy Controls in Information Systems and Organizations. National Institute of Standards and Technology, 2022.
53. NIST Special Publication 800-53 Rev. 5. Security and Privacy Controls for Information Systems and Organizations. National Institute of Standards and Technology, 2020.
54. NIST Special Publication 800-63B. Digital Identity Guidelines: Authentication and Lifecycle Management.
55. NIST Special Publication 800-81-2. Secure Domain Name System (DNS) Deployment Guide.
56. NIST Special Publication 800-86. Guide to Integrating Forensic Techniques into Incident Response. National Institute of Standards and Technology, 2006.
57. NIST Special Publication 800-92. Guide to Computer Security Log Management. National Institute of Standards and Technology, 2006.
58. NIST Special Publication 800-94 Rev. 1 (Draft). Guide to Intrusion Detection and Prevention Systems (IDPS).
59. OASIS. STIX Version 2.1. Structured Threat Information Expression. OASIS Open, 2021.

60. OASIS. TAXII Version 2.1. Trusted Automated Exchange of Intelligence Information. OASIS Open, 2021.
61. Open Container Initiative. OCI Image Format and Runtime Specifications.
62. Open Information Security Foundation. Suricata User Guide.
63. Open Networking Foundation. OpenFlow Switch Specification.
64. PCI Security Standards Council. Payment Card Industry Data Security Standard (PCI DSS), Requirements and Testing Procedures, Version 4.0.
65. RFC 2131. Dynamic Host Configuration Protocol. IETF, 1997.
66. RFC 2865. Remote Authentication Dial In User Service (RADIUS). IETF, 2000.
67. RFC 4033-4035. DNS Security Introduction and Requirements (DNSSEC). IETF, 2005.
68. RFC 5424. The Syslog Protocol. Internet Engineering Task Force (IETF), 2009.
69. RFC 7011. Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of Flow Information. Internet Engineering Task Force (IETF), 2013.
70. SANS Institute. Common and Best Practices for Security Operations Centers: Results of the 2023 SOC Survey.
71. Splunk Docs. Search Processing Language (SPL) Reference. Splunk Inc. URL: <https://docs.splunk.com>
72. Splunk Docs. Splunk Enterprise and Splunk Enterprise Security Documentation. Splunk Inc. URL: <https://docs.splunk.com>
73. The Institute of Internal Auditors. International Professional Practices Framework (IPPF).
74. The Institute of Internal Auditors. The IIA's Three Lines Model: An Update of the Three Lines of Defense. IIA, 2020.
75. The Zeek Project. Zeek Documentation: Introduction and Architecture.