Бойко Віктор Дмитрович
к.т.н., доцент, доцент кафедри кібербезпеки,
Національний університет «Одеська юридична академія»

ВИКЛИКИ УПРАВЛІННЯ КІБЕРБЕЗПЕКОЮ МЕРЕЖЕВИХ СИСТЕМ ТА ІДЕНТИФІКАЦІЇ ВТОРГНЕНЬ ПІД ЧАС КРИЗОВИХ ПОДІЙ

DOI https://doi.org/10.36059/978-966-397-537-5-2

1.1. Вступ

Починаючи з ARPANET, системи інформаційно-комунікаційних систем (information communication systems – ICS) спочатку проєктувалися, як стійкі, гнучкі системи націлені на максимальне збереження функціональності в умовах несприятливих впливів і факторів різного роду.

Будь-яка сучасна інформаційно-комунікаційна мережа (information communication network – ICN) з доступом до глобального інфопростору практично безперервно піддається атакам різного роду і характеру [1], та несприятливим впливам і вражаючим факторам. Сучасні ICN апріорі є вразливими – що підтверджується довгим списком прецедентів, описаних у [2]. Практика захисту та відновлення функціонування ICN показує, що загальна тенденція до ускладнення, централізації, ієрархізації та недостатньо продуманих рішень при плануванні архітектури глобальних ICN призвели до того, що сучасні ICN втратили істотну частину своєї гнучкості та здатності протистояти зовнішнім та внутрішнім впливам.

Ускладнення також призвело до того, що на сучасному етапі однією зі складних проблем захисту ICN є не максимізація стійкості ICN до несприятливих впливів і вражаючих факторів, а мінімізація часу відновлення функціональності та працездатності ICN після інциденту. Найбільш показовим є випадок «Київстар», який зміг відновити повну функціональність своєї мережі лише через 7 днів (з 12 до 20 грудня 2023 року [3; 4]).

У архітектурі безпеки ICN веб-сервери відіграють ключову, детермінуючу роль, оскільки вони є невід'ємними компонентами, що забезпечують безперебійне функціонування критично важливих

інформаційно-комунікаційних послуг. Вони є вузлами обробки запитів, сховищами даних та платформами для доступу до широкого спектру додатків та сервісів, від фінансових транзакцій до управління муніципальною інфраструктурою. Вихід з ладу цих критичних елементів призводить до каскадних наслідків, що поширюються по всій системі, мультиплікуючи збитки та провокуючи вторинні порушення функціональності, що ускладнює загальне відновлення.

Приклад цього явища ілюструється впливом збоїв мобільної мережі на банківський сектор, де значна частина банкоматів, POS-терміналів та систем двофакторної автентифікації залежать від функціонуючих веб-серверів операторів зв'язку. Втрата одного ключового елемента інфраструктури може призвести навіть до паралічу фінансових послуг та порушення роботи критичних комунальних систем, таких як вуличне освітлення.

Таким чином, підвищення живучості веб-серверів є критично важливим для забезпечення стійкості всієї мережевої інфраструктури. Це потребує інтегрованого підходу, що включає технічні рішення, організаційні заходи та постійне вдосконалення методологій, спрямованих на витримування зростаючих кіберзагроз та забезпечення безперебійного функціонування в кризових умовах.

З огляду на експоненційне зростання кількості та складності кібератак, включаючи такі загрози як руткіти, програми-вимагачі та цілеспрямовані атаки на промислові системи, захист веб-серверів є невід'ємною передумовою забезпечення живучості мережі. Це вимагає не лише впровадження передових технологій виявлення вторгнень та забезпечення цілісності даних, але й глибокого аналізу взаємозв'язків між мережевими компонентами та прогнозування потенційних каскадних відмов.

1.2. Загальні виклики стабільності мережі: погляд згори

1.2.1. Вразливість ICN у кризових умовах

Останнім часом почастішали збої глобальних інформаційних мереж. На думку авторів, це є наслідком кількох причин.

По-перше, спостерігається вибухове зростання обсягу та складності ICN. Forbes наводить наступні дані щодо мобільних операторів

в Україні: 19 мільйонів абонентів у Vodafone [5], 8,9 мільйонів у Lifecell [6], 24 мільйони у Київстар [7]. За цими ж даними Lifecell має близько 9000 базових станцій.

По-третє, досвід безперебійної роботи мобільних та інтернет-мереж призвів до того, що багато різних служб так чи інакше використовували їх як частину своєї інфраструктури. Насамперед це стосується банкінгу – банкомати, термінали поповнення, POS-термінали здебільшого використовують мобільний зв'язок для комунікації з банком і при виході з ладу мобільної мережі перестають функціонувати. Сюди ж відноситься широко розповсюджена та просувана як «передова технологія безпеки» двофакторна автентифікація, при якій підтвердження особи користувача відбувається за допомогою SMS-повідомлень. Згідно з джерелами Forbes, у ПриватБанку під час відключення «Київстар» не працювало до третини POS-терміналів та близько 5 % банкоматів. Це найбільш поширені випадки, однак, глобальний збій зв'язку показав, що існують й інші вразливі системи. Зокрема, ЛКП «Львівсвітло» було змушене здійснювати відключення ліній вуличного освітлення в ручному режимі [8].

У роботі [9], що вийшла ще до початку повномасштабного вторгнення, робився наступний висновок:

«В умовах впливу НВ важливим стає не тільки управління ліквідацією наслідків безпосередніх загроз, а й боротьба за функціональну живучість з метою не допустити розвалу системи. В результаті НВ системи можуть безпосередньо виходити з ладу не тільки системи, а й породжуватися»вторинні ефекти», впливаючи на суміжні з ними системи. Для загального розвалу інформаційної системи в результаті НВ часто не обов'язково, щоб було виведено 100 % її підсистем, а досить просто створити зниження працездатності одного або декількох ключових компонентів системи».

Подальші події (блекаути, втрата функціональності систем через бойові дії, нещодавнє відключення мобільного зв'язку «Київстар») підтвердили актуальність викладених у статті положень.

Все перелічене підкреслює актуальність проблеми забезпечення живучості сучасних ICN та гостро ставить питання про надійність і живучість інфраструктурних систем.

1.2.2. ICN та ICS: еволюція загроз та приклади вторгнень

Як зазначалося вище, більшість існуючих інформаційно-комунікаційних систем (ICN) після введення в експлуатацію піддаються атакам різного роду та характеру [1] і з великою ймовірністю можуть бути зламані або виведені з ладу. У випадку, якщо зламу або іншим несприятливим впливам піддаються глобальні інформаційні системи, або інформаційні системи, що обслуговують індустріальні та промислові комплекси (industrial control system – ICS), ризики та масштаби збитків багаторазово збільшуються.

Донедавна основним фокусом небезпеки були розробники промислових та інфраструктурних інформаційних систем, які, на відміну від розробників ПЗ «загального призначення», запізнюються з реакцією на нові загрози. Наприклад, при аналізі безпеки ICS багато фахівців схильні покладатися на так званий "air gap" – «повітряний зазор», що ізолює ICS від глобальних інформаційно-комунікаційних систем. Простіше кажучи – якщо керуюча система не підключена до Інтернету – звідти не може здійснюватися атака. При цьому один із найперших прикладів атаки – вірус Stuxnet, якраз був побудований на подоланні «повітряного зазору» [10].

При аналізі інцидентів мережевої інфраструктури найчастіше посилаються на аналіз розповсюдження та шкоди, завданої Stuxnet [10] – мережевого хробака, націленого на руйнування центрифуг фірми Siemens, задіяних у ядерній програмі Ірану.

Цей хробак розповсюджувався мережею, використовуючи зараження флеш-накопичувачів та чотири вразливості категорії **zero-day**. Наприклад, зараження машини відбувалося не за допомогою традиційного запуску autorun.inf, яке відоме всім і досить легко відстежується та блокується засобами операційної системи, а за допомогою .LNK-вразливості, яку складно виявити та ліквідувати.

Розповсюдження хробака відбувалося доти, доки він не опинявся на машині, яку ідентифікував як потрібну йому (тобто SCADA-систему Siemens, що має певну конфігурацію). Після цього відбувалося перехоплення управління та виконання руйнівних дій. Цікаво, що таким чином (приховано заражаючи флешки) хробак зміг дістатися до комп'ютерів ядерної програми, які не мали виходу в Інтернет та вважалися захищеними від зовнішніх атак.

Ще одним відомим прикладом цілеспрямованої шкідливої програми, націленої на ураження об'єктів промислової інфраструктури, є вірус **Triton** [11; 12], який був націлений на атаку «останнього рубежу оборони» – виведення з ладу приладових систем безпеки (SIS) Schneider Triconex.

Найбільш відомими прикладами атак в Україні можуть слугувати епідемії **WannaCry**, **Petya**, а також збої української енергосистеми у 2015 році, описані в [13] та в [14].

Про готовність комп'ютерної інфраструктури свідчить динаміка зараження під час епідемій програм-вимагачів Petya та WannaCry [15], коли більше третини комп'ютерних мереж (включаючи банківські та державні) виявилися виведеними з ладу. Загальний стан справ з еволюцією атак на об'єкти інфраструктури докладно розглянуто в [16] та [17].

Також одним з різновидів програм-вимагачів є шифрувальник **Snake**, виявлений компанією FireEye (див. [18]), націлений на вибіркову атаку промислових об'єктів, які використовують обладнання General Electric. Наприклад, програма-вимагач атакувала сервер GE Digital Proficy Process [19].

При цьому експерти FireEye стверджують, що якщо до цього моменту шифрувальники були націлені «в простір» – тобто заражали будь-який доступний користувацький комп'ютер, то у випадку Snake можна говорити про цілеспрямовану, таргетовану загрозу конкретним об'єктам інфраструктури.

1.2.3. Аналіз відновлення працездатності ICN: вторинні ефекти

Процес відновлення працездатності ICN являє собою поєднання різних факторів, що часто слабо піддаються оцінці. При відновленні працездатності обслуговуючий персонал може стикатися з проблемами різної природи та генези – недоліком запасних частин, дефіцитом дублюючих потужностей тощо, тощо. Можуть виникати погано передбачувані вторинні ефекти, які мультиплікують загальні збитки та викликають вторинні за своєю природою збої в роботі, що ще більше ускладнює відновлення системи. Це особливо характерно для так званих мереж масового обслуговування, окремим випадком яких є ICN.

Приклад такого вторинного ефекту виникає в таких мережах при виході з ладу деякої кількості вузлів мережі. Оскільки мережа є гнучкою за своєю природою, то вихід одного або декількох вузлів не призводить до загального виходу з ладу мережі безпосередньо, однак, призводить до того, що користувачі, які втратили точки обслуговування, перенаправляють свої запити на інші вузли мережі. Загальне навантаження на ці вузли зростає, що насправді є «мирним» варіантом атаки на відмову (DDoS атаки). Якщо мережа спроєктована без урахування цього ефекту, збільшення навантаження може призвести до відмови нових вузлів, що вціліли при первинному впливі. Вторинні відмови, у свою чергу, призводять до того, що зростає навантаження на вузли мережі, що залишилися – що веде до нової серії відмов. І так далі – процес може розвиватися аж до повної відмови мережі. У разі, якщо несприятливим впливам піддаються інформаційні системи, що обслуговують індустріальні та промислові комплекси, ризики та масштаби збитків багаторазово збільшуються [9].

У статті [2] було описано модель живучості мережі, яка ґрунтується на представленні мережі у вигляді сукупності віртуального або реального обладнання, які в рамках завдання розглядаються як граф вузлів, пов'язаних з'єднаннями різного характеру. Ці зв'язки між елементами можуть розглядатися як зв'язок по ресурсу, що відповідає категоріям живучості системи («енергія» – «інформація» – «речовина»), а сама модель має структурно/функціональний акцент («мережевий» за термінологією [1]). Також у моделі передбачено використання спеціальних, «віртуальних» вузлів, які моделюють причинно-наслідкові залежності та відносини між компонентами [20]). Це дозволяє доповнити модель елементами багатофакторного аналізу. Кожен із вузлів системи може бути частиною ICN, її елементом, або окремим робочим компонентом робочої підсистеми. При цьому цей вузол в рамках моделі має характеристики, що включають час і характер відновлення елемента. У статті пропонувалась методика оцінки часу відновлення використовує безрозмірні позамасштабні оцінки «важливості», «уразливості» елемента з точки зору загальної задачі мінімізації часу відновлення системи в цілому. Така оцінка традиційно спирається на критерій Бірнбаума [21] та ідеї, викладені в [22].

У роботі [22] розглянуто модель відновлення функціональності системи, в якій функція відновлення працездатності описана за допомогою диференціальних рівнянь. Рішення таких рівнянь для інтервалу часу, меншого, ніж чверть періоду рішення (0; π/2), в залежності від отриманих коефіцієнтів можна розділити на три основні категорії, кожна з яких відповідатиме режиму відновлення добре підготовленої, середньопідготовленої та поганопідготовленої системи. При цьому добре підготовлена система описується експоненційним законом, середньопідготовлена система – лінійним, погано підготовлена – тригонометричним – не дуже добре підготовлена система. При цьому під тригонометричним законом мається на увазі залежність, що відповідає першій чверті повного періоду синусоїдальної функції – на ділянці (0; π/2). Істотним моментом використання такої моделі є можливість як оцінки часу відновлення, і розгляд різних сценаріїв розвитку подій – зокрема каскадного виходу мережі з ладу, описаного вище.

Ми пропонуємо доповнити модель, щоб розширити її можливості щодо оцінки вторинних сценаріїв та наслідків кіберзагроз, додавши фактор контрольованого ризику виходу з ладу окремих компонентів системи. Для цього характеристики вузлів когнітивно-імітаційної моделі доповнюються спеціальною характеристикою, яка моделює ризик виходу з ладу компонента на кожному кроці моделювання процесу усунення несправностей та відновлення працездатності системи. Ця величина в залежності від цілей і завдань моделювання може характеризуватись різними за характеристиками розподілу (Нормальне, Вейбулла різних порядків), а в спрощеній моделі може бути представлена математичним очікуванням.

1.2.4. Оптимізація відновлення працездатності ICN: принципи ефективного резервного копіювання веб-серверів

Як зазначалось вище, на сучасному етапі, основною проблемою часто є не максимізація стійкості ICS, а мінімізація часу відновлення функціональності та працездатності ICS після інциденту [23, 1]. Тому на перший план виходить раціональна організація систем резервного копіювання та відновлення ICS (бекапів – backups). Одним із важливих елементів такого резервування є резервування інформації веб-серверів, причому у широкому розумінні цього

терміна – веб-сервер, база даних, операційна система, програмне забезпечення тощо. Нижче наведено рекомендації в організації системи ефективних бекапів, які розроблені в основному для веб-проєктів і систем, які надають інфраструктуру для цих проєктів, проте, ці рекомендації є порівняно універсальними і можуть бути використані для організації резервування більшості проєктів, так чи інакше пов'язаних з ICS.

Розглянуті системи можуть сильно відрізнятися за масштабом та задіяними ресурсами – залежно від проєктів. Різні проєкти вимагають різних стратегій відновлення, різної частоти збереження резервних копій тощо. Тому на попередньому етапі потрібен збір інформації та всебічне дослідження проєкту. Які саме частини проєкту є важливими та незамінними? Автор стикався із ситуацією, коли персонал резервував операційну систему цілком, не розуміючи, що саме треба зберігати на диск. Наскільки важливою є безперебійна робота проєкту? Який проміжок часу є на відновлення у найгіршому випадку?

Існує кілька основних принципів, що склалися в процесі експлуатації ICS та пройшли перевірку практикою:

1. Робоча система та система, що зберігає резервні копії, повинні бути розділені фізично.
2. Резервні копії проєкту мають шифруватися.
3. Після розгортання системи резервного копіювання має бути проведене хоча б одне тестове відновлення проєкту із резервної копії.
4. Запуск та всі проміжні етапи резервування повинні здійснюватися автоматично за розкладом. Процес резервування має документуватись у системних журналах.
5. Залежно від проєкту має зберігатися декілька копій. Повинен бути організований процес їхньої ротації.

Розглянемо ці принципи докладніше.

1. Робоча система та система, що зберігає резервні копії, повинні бути розділені фізично.

Резервні копії не повинні зберігатися на робочій станції проєкту. Це має бути окрема станція, дуже бажано фізично розділена з робочою. Різні приміщення, в ідеалі – різні будівлі. Ця вимога обумовлена здоровим глуздом. Якщо резервна копія зберігається на

робочій станції, то при виході станції з ладу доступ до бекапу буде втрачено. Фізичний поділ працює на додаткову безпеку бекапу. Слід пам'ятати, що крім кібераттак, робоча станція може піддаватися суто фізичним впливам: пожежа, затоплення приміщення, пошкодження внаслідок бойових дій. У цьому випадку бекап збережений на віддаленій дистанції має набагато більше шансів зберегтися, ніж якщо він буде в одному приміщенні з робочою станцією.

У межах невеликих та середніх проєктів реалізація цього принципу, всупереч першому враженню, необов'язково потребує ресурсів та витрат праці. Залежно від вимог до проєкту, в ролі сервера бекапу може виступати окремий комп'ютер з розгорнутим на ньому ssh-сервером – персональний комп'ютер, одноплатний комп'ютер. Автори зустрічали випадки організації систем бекапу на базі таких зовні «несерйозних» станцій, як списані ноутбуки та роутери з кастомною прошивкою. Повторимося – все залежить від вимог проєкту.

2. Резервні копії проєкту мають шифруватися.

Ця умова випливає із першого принципу. Бекап – це інформація про проєкт. Часто із конфіденційними даними. Якщо бекап зберігається окремо від робочого проєкту – у зловмисників з'являється потенційна можливість доступу до інформації в обхід систем захисту основного проєкту. Тому бекап, що зберігається, повинен бути захищений від доступу третіх осіб, як мінімум засобами сильної криптографії – тобто зашифрований.

3. Після розгортання системи резервного копіювання має бути проведене хоча б одне тестове відновлення проєкту із резервної копії.

Резервування часто виконується для досить складних структурою і великих за обсягом даних проєктів. У цьому випадку перевірити можливе відновлення зі збереженого бекапу чи ні, можна тільки проробивши цю процедуру на практиці. Інакше виникає ситуація, що отримала у фахівців назву «бекапу Шредінгера» – система резервування начебто працює, бекапи зберігаються, але чи все буде відновлено після збою, ніхто не знає.

Ще один неочевидний момент виникає тоді, коли проєкт змінює робочу конфігурацію, а бекапи зберігаються виходячи зі старої. Як було сказано вище, щоб така ситуація не виникала – потрібно щонайменше одноразове перевірочне відновлення з резервної копії. Бажано,

щоб воно проводилося регулярно і точно, воно вимагає повторення при кожній зміні конфігурації проєкту.

4. Запуск та всі проміжні етапи резервування повинні здійснюватися автоматично за розкладом. Процес резервування має документуватись у системних журналах.

Якщо процес резервування зав'язаний на людину – виникає людський фактор. Люди схильні плутати необхідні дії, забувати, хворіти тощо. Навіть якщо людині для збереження бекапу потрібно натиснути одну кнопку – велика ймовірність, що вона її забуде натиснути саме в день, коли знадобиться бекап.

Тому слід виключити людський чинник із процесу збереження бекапів. Навіть якщо не розглядати спеціалізоване стороннє програмне забезпечення, будь-яка сучасна операційна система має вбудовані можливості, які дозволяють планувати та здійснювати запуск програм на регулярній основі. Для ОС сімейства Windows – це task sheduler («Планувальник завдань»), для POSIX-сумісних систем (Linux, BSD) – це cron, або таймери служби systemd у більш сучасному варіанті.

Автоматизація процесу дозволяє отримати ще одну перевагу – процес резервування може запускатися у позаробочий час (наприклад о 3 або 4 годині ночі), не заважати робочим процесам та отримуючи максимальний доступ до ресурсів самостійно.

Однак, як у будь-якій іншій системі, автоматизація не завжди спрацьовує гладко спочатку, тому вона повинна обов'язково супроводжуватися звітами про минулі успішні бекапи, щоб це можна було проконтролювати.

5. Залежно від проєкту має зберігатися декілька копій. Повинен бути організований процес їхньої ротації.

Бекап рідко буває поодиноким. Залежно від проєкту, бекап може бути досить об'ємним. Це вимагає місця для зберігання та ефективного поводження з резервними копіями, що накопичуються. Оптимально організовувати стек та ротацію бекапів, а також зберігати бекапи за системою «рік – місяць – день тижня». Це дозволить з одного боку ефективно використовувати доступні ресурси зберігання даних, а з іншого – не втратити інформацію при збоях в роботі основної станції.

Дотримання перерахованих вище принципів дозволить організувати стійку і надійну систему резервування проєкту та уникнути втрат при можливих збоях, чим суттєво підвищить стійкість до несприятливих впливів та вражаючих факторів різного роду.

1.3. Питання безпеки веб-серверів як складової загальної мережевої стійкості

1.3.1. Веб-сервери, як основна мішень атак ботнетів

Веб-сервери є основною мішенню для атак, що здійснюються за допомогою ботнетів. Ця тенденція обумовлена їхньою критичною роллю у функціонуванні мережевої інфраструктури та широким розповсюдженням.

У широкому розумінні, веб-сервер охоплює не лише безпосередньо програмне забезпечення (наприклад, стеки LAMP/LAPP), але й усю пов'язану інфраструктуру. Він є основою та ключовим компонентом сучасної всесвітньої мережі.

Окрім очевидних великомасштабних рішень, значного поширення набули також різноманітні реалізації серверів для домашніх мереж, а також сервери, що забезпечують веб-інтерфейс для різних застосунків. Зокрема, у системах Linux веб-сервер використовується для конфігурації системи друку Common UNIX Printing System (CUPS). Широке впровадження цих рішень сприяло повсюдному поширенню веб-серверів, водночас перетворивши їх на найбільш очевидну мішень для кібератак.

Можна виділити наступні причини, з яких веб-сервери є бажаними та пріоритетними цілями для кібератак:

– Вебсервер за своєю природою є відкритою системою, яка постійно підключена до зовнішніх інформаційно-комунікаційних мереж. Відповідно, за невеликим винятком, для атаки на нього не потрібен фізичний доступ.

– З широким розповсюдженням веб-інтерфейсів веб-сервери набувають все більшого поширення та можуть інтегруватися у найнесподіваніші місця, наприклад, у системи конфігурації принтерів та іншого офісного обладнання.

– Веб-сервери все частіше виконують роль інтегруючого рішення для організації різноманітних систем колективної взаємодії

на рівні як зовнішніх зв'язків, так і роботи в інтранеті (внутрішніх локальних мережах організацій), а також функціонування промислових систем, які мають бути відокремлені від зовнішнього світу, проте на практиці ця умова рідко дотримується.

У разі успішної атаки шкідливе програмне забезпечення має високу ймовірність залишатися невиявленим протягом тривалого часу.

– Шкідливе програмне забезпечення, що впровадилося на сервер, потенційно здатне інфікувати велику кількість комп'ютерів, що взаємодіють з ним, за короткий термін.

– Шкідливе програмне забезпечення, що впровадилося на сервер, потенційно отримує доступ як до конфіденційної інформації на самому сервері, так і за його межами.

– Персонал, відповідальний за встановлення та налаштування веб-систем, часто є некомпетентним та некваліфікованим.

– Попередній факт ускладнюється тим, що існує безліч різних реалізацій програмного забезпечення веб-серверів, конфігурування та забезпечення безпеки яких можуть суттєво відрізнятися, що ускладнює навчання та перехід від однієї конфігурації до іншої.

– Функціонування веб-сервера забезпечується набором (стеком) взаємодіючих програм: власне веб-сервер, база даних, мова програмування та пов'язані з нею фреймворки, системи конфігурації та взаємодії, операційна система та протоколи доступу до неї. Кожен із цих компонентів, за умови неграмотного конфігурування та експлуатації, може становити потенційну або реальну вразливість.

– Зі складністю та різноманітністю систем роботи веб-сервера пов'язаний великий обсяг «сирої» інформації, в якій легко загубитися, якщо не володіти технічними засобами аналізу та обробки. Текстові логи відвідувань сервера за кілька днів цілком можуть досягати розміру порядку гігабайтів, що робить безглуздим їх «неозброєний перегляд».

Таким чином, забезпечення безпеки веб-серверів на всіх рівнях їх функціонування є і буде актуальним та важливим завданням, враховуючи ключову роль, яку веб-сервери відіграють у структурі сучасного інформаційного простору, а також все зростаючу роль інформаційних комунікацій (у тому числі через епідемію коронавірусу та пов'язане з нею підвищення інтересу до дистанційних форм роботи).

1.3.2. Руткіти як головний інструмент розповсюдження ботнетів

Основним джерелом мережевих атак в даний час є ботнети – мережі заражених malware комп'ютерів, що мають доступ до інтернету. Ботнети є серйозною проблемою – досить сказати, що більшість комп'ютерних атак здійснюється за їх допомогою [24; 25].

Одним з найбільш небезпечних різновидів шкідливого програмного забезпечення є руткіти (rootkit). Так називають ПЗ, націлене на приховане впровадження в систему (в даному випадку – веб-сервер) та отримання в ній прав адміністратора (ескалація привілеїв). Руткіт прагне залишатися прихованим, але при цьому йому, як правило, потрібен зв'язок із зовнішнім світом [26].

Ранні дослідники класифікували руткіти та аналогічне за призначенням ПЗ за класами прихованості [27]:

– **Програмне забезпечення, що демаскує себе – Рівень (–1)**: зараження тягне за собою порушення функціональності зараженого об'єкта, що робить виявлення зараження неминучим.

– **Відсутність прихованості – Рівень 0**: жодних спеціальних заходів для приховування присутності інфекції не вживається.

– **Елементарна прихованість – Рівень 1**: виконуються основні дії щодо запобігання виявленню, такі як збереження у змінюваних файлах метаданих дати та часу, щоб приховати факт їхньої зміни.

– **Середній ступінь маскування – Рівень 2**: частина образу (image) зараженого об'єкта в його початковому чистому стані до зараження зберігається для «показу» системі, щоб допомогти приховати «слід» вірусу.

– **Просунуте маскування – Рівень 3**: додаток використовує методи маскування, спеціально призначені для приховування зараження від спеціальних додатків, що забезпечують безпеку.

На ранній стадії основним об'єктом атак були фінансові та платіжні системи: підробка реквізитів банківських кредитних карток, злом кодувань супутникових телевізійних каналів тощо.

У міру збільшення рівнів прихованості впроваджуваний шкідливий код все більше адаптується не з метою «традиційного» вандалізму (видалення, псування файлів тощо), скільки для здійснення прихованої діяльності.

Такою діяльністю може бути:

– приховане зараження комп'ютерів, що контактують із сервером;
– збір конфіденційної інформації;
– організація та управління атаками.

На наступному етапі зломи стали інтелектуальнішими – зокрема, злам мережі банкоматів з метою отримання грошей та програм-вимагачів (ransomware) – шкідливих додатків, що шифрують диск та пропонують перевести гроші для дешифрування.

Наприклад, відомий мережевий хробак Stuxnet приховано розповсюджувався по системах, приховано заражаючи комп'ютери та уникаючи виявлення антивірусними системами доти, доки конфігурація зараженого комп'ютера не збіглася з потрібною. Після цього хробак спрацював та вивів з ладу центрифуги фірми Siemens, керуючі комп'ютери яких навіть не були підключені до інтернету (хробак потрапив туди із зараженого флеш-носія) [10].

Справжньою проблемою є атаки, що здійснюються для отримання приватної інформації, яка не підлягає розголошенню (наприклад, після атаки на Korea Hydro & Nuclear Power Co., Ltd. (KHNP) зловмисники отримали доступ до креслень та інструкцій для атомних реакторів [28]).

Все вищеперелічене визначає труднощі у виявленні не тільки діяльності руткіта, але часто і самого факту його вторгнення. Що опосередковано підтверджується наступними цифрами.

Згідно з Mandiant Security Effectiveness Report 2020 [19], 53% кібератак закінчуються невиявленим впровадженням в організаційні структури, і 91% всіх інцидентів не викликають спрацьовування систем безпеки.

У 2015 році середній час виявлення факту успішної кібератаки становив від 50 до 70 днів [29].

Згідно зі звітами IBM [30], тільки на ідентифікацію витоків йде до 280 днів, оцінки інших експертів варіюються в межах від сотень днів до року.

1.3.3. Аудит веб-серверів та виявлення фактів вторгнення у систему

Комп'ютер, підключений до всесвітньої павутини, безперервно піддається атакам та «випробуванню на міцність» його систем

захисту. При цьому кількість ботнетів, заражених комп'ютерів та атак зростає з кожним роком. Крім того, програмне забезпечення, техніки, тактики та стратегії сторони, що атакує, постійно вдосконалюються.

Визначальним фактором успішного функціонування сучасних ботнетів є часовий інтервал між вторгненням та зараженням комп'ютера та виявленням того, що це сталося. Цей інтервал визначає ступінь шкідливості malware – чим він довший, тим більше атак встигне здійснити заражена машина, до того, як це виявиться і будуть застосовані заходи у вигляді налаштування фаєрволу, антивірусного сканування тощо. аж до переустановки операційної системи. Однак, виявлення вторгнень у систему в сучасних умовах все ще недостатньо швидке. У 2015 році середній час виявлення факту успішної кібератаки складав від 50 до 70 днів [29]. Згідно з звітами IBM за 2020 рік [30], на ідентифікацію витоків витрачалося в середньому 280 днів, Madiant Security Effectiveness Report 2020 [19] вказує, що 53 % кібератак закінчується працювання систем безпеки.

Проблему намагаються вирішити системним підходом – використанням технології SIEM (Security information and event management), яка є синтезом існуючих раніше систем SEM (Security Event Management) і SIM (Security information management) [31; 32]. Однак, на практиці впровадження складних, витратних за ресурсами та потребують кваліфікації керуючого персоналу систем захисту, часто виявляється утрудненим для розробників та адміністраторів ICN середньої та нижчої ланки.

При цьому робота з ключовою для виявлення та управління кібербезпекою інформацією донедавна будувалася виходячи з парадигми того, що вся інформація в ICN буде доступна в будь-який момент часу, що функціонування ICN мінімально схильне до збоїв і без урахування зростання складності ICN. Ці питання докладно розглядалися у роботі [9] що вийшла ще до початку повномасштабного вторгнення, блекаутів та кількох глобальних відключень важливих структурних складових ICN («Київстар», facebook, microsoft тощо). Наступні події (блекаути, втрата функціональності систем через бойові дії, нещодавнє відключення мобільного зв'язку «Київстар») підтвердили актуальність викладених у статті положень та були додатково розглянуті у [2]. Сучасна ICN існує в умовах

все зростаючої щільності несприятливих впливів і факторів, що вражають [1]. При цьому структура самої ICN забезпечує досить великий ступінь автономності, що з одного боку дозволяє швидше відновлювати систему після збоїв або вимушеного простою (наприклад, в результаті блекауту), а з іншого ускладнює управління інформацією та ключовими подіями для кібербезпеки ICN. Наприклад, якщо блекаут викликав втрату зв'язності окремих частин ICN, це може призвести до збоїв у роботі SIEM, яка не розрахована на несинхронну роботу з розподіленою системою.

1.3.4. Засоби виявлення та пошуку руткітів

Класичний спосіб боротьби та виявлення шкідливого ПЗ полягав у побайтовому аналізі бінарних файлів з метою виявлення порушень цілісності файлу, невірних значень його метаданих. Пізніше до цього арсеналу додалися методи виявлення відомих версій шкідливого ПЗ за їхніми характерними сигнатурами (фрагментами коду), ще пізніше – евристичний аналіз файлів на пошук невідомих шкідливих додатків за характерними ознаками поведінки.

Серед сучасних засобів виявлення та пошуку руткітів в рамках локальної системи можна виділити три основні категорії:

– аналіз бінарних файлів;
– аналіз поведінки системи та користувача;
– аналіз системних журналів.

Традиційний класичний аналіз бінарних файлів системи, який, своєю чергою, включає сканування з метою виявлення сигнатур відомих шкідливих програм та евристичний аналіз. Як пошук сигнатур, так і евристичний аналіз, за всієї своєї практичної цінності, мають наступні недоліки.

Аналіз сигнатур, з одного боку, простий, інтуїтивно зрозумілий, не вимагає великого обсягу обчислювальних ресурсів і дозволяє визначати конкретну атаку (та використане для неї шкідливе ПЗ) з високою точністю та відносно низьким рівнем хибних спрацьовувань (false positives). З іншого боку, «на довгій дистанції» збір сигнатур та визначення вирішальних правил виявляється гранично трудомістким завданням – сучасні бази даних, наприклад, у базі даних антивірусу Comodo міститься понад 86 359 675 сигнатур – майже сто мільйонів.

Пошук та виявлення, підтримання актуальності та оновлення таких баз є трудомістким завданням. Сканування системи також має тенденцію затягуватися – як через зростання баз даних із сигнатурами, так і через зростання обсягів сканованого програмного забезпечення. Ця проблема може частково зніматися зростанням швидкодії системи та роботою з вибіркового сканування найбільш вразливих місць системи, однак швидкість роботи антивірусу стає все повільнішою. Крім того, пошук за сигнатурами не спрацьовує на поліморфних та самомодифікованих вірусах, на вірусах, що шифрують своє тіло тощо. Він не виявляє нових екземплярів шкідливого ПЗ і не здатний запобігти атаці з використанням щойно написаного програмного забезпечення.

Евристичний аналіз є в деякому сенсі протилежною аналізу сигнатур процедурою пошуку. Евристики для пошуку шкідливого ПЗ можуть виявляти нові та щойно написані екземпляри, база даних із правилами менш громіздка, ніж база із сигнатурами, відповідно, перевірки можуть теоретично відбуватися швидше та з меншими витратами ресурсів. Однак евристичні аналізатори зазвичай відрізняються високим рівнем хибних спрацьовувань. Евристичні правила досить легко обходяться, якщо вони відомі розробнику шкідливого ПЗ заздалегідь. Поширена практика, коли такі розробники заздалегідь закладають у ПЗ атаки спеціальні прийоми та системи, які дозволяють обходити евристичний аналіз – наприклад, фрагменти коду, що ламають або виводять з ладу відладчик при спробі трасування результатів виконання програми.

Як правило, існуюче антивірусне ПЗ використовує комбінований підхід, що поєднує пошук за сигнатурами та евристичний аналіз файлів. Однак ефективність такого ПЗ далека від заявленої.

Другий тип пошуку шкідливого ПЗ орієнтований на непрямий аналіз того, що відбувається в системі. Такий аналіз може включати аналіз мережевої активності (зміст та характер вихідного та вхідного трафіку), аналіз активності додатків, аналіз активності користувача.

Наприклад, найпростішим видом такого аналізу може бути аналіз стану портів системи, оскільки частина шкідливого ПЗ може використовувати прослуховування вільних портів для комунікації із зовнішнім світом.

Аналіз такого типу може бути досить ефективним, однак широке розповсюдження піринговых та криптографічних технологій комунікацій все більше ускладнює його роботу, оскільки все складніше стає аналізувати трафік, який не просто шифрується, а набуває такої властивості, як однорідність (наприклад, у разі використання технологій Tor та I2P – складно відокремити в потоці трафіку, що проходить через систему, нативний трафік самої системи від потоку транзитних даних, які проходять крізь систему, оскільки вона віддає частину своїх ресурсів загальній мережі).

З урахуванням усіх перелічених переваг та недоліків проблема захисту системи від вторгнень продовжує залишатися актуальною. При такому підході більша частина безпеки сервера визначається профілактичними заходами (hardening) та грамотною конфігурацією системи – встановленням брандмауера, використанням стійких паролів та системи відкритих ключів, правильним розподілом ролей та прав системних користувачів, зміною налаштувань за замовчуванням тощо.

Якщо розглядати існуючі технології виявлення вторгнень у сучасних інформаційно-комунікаційних мережах на більш високому рівні, у рамках технології SIEM для локальної мережі на рівні офісу, підприємства або технологічної мережі, то також можна виділити кілька основних категорій, які використовують різні методи виявлення. У найзагальнішому вигляді до них можна віднести:

– аналіз мережевого трафіку;
– аналіз вмісту пам'яті та жорстких накопичувачів;
– аналіз системних журналів.

Зазначимо, що системи виявлення вторгнень найчастіше будуються за гібридною моделлю, використовуючи кілька різних технологій.

Якщо розглядати ці схеми з точки зору виявлення джерела malware, то перша з цих технологій потребує серйозної підготовки та доступу до всіх ланок ієрархії ICN, через які проходить атака. При цьому аналіз мережного трафіку серйозно утруднений через фрагментацію ICN, яка у свою чергу пояснюється відставанням у впровадженні технологій IPv6, через що сучасний інфопростір є ієрархією підмереж з «сірими» IP-адресами, в якій існує багато рівнів

вкладеності і часто – підключення, що динамічно змінюються (найочевидніший приклад – смартфон, що перемикається між WiFi-мережами). Все це сильно ускладнює детектування та виявлення джерела атаки шляхом аналізу мережевого трафіку. Додатковими факторами, що погіршують, є все зростаючий обсяг переданих даних, низька грамотність як технічного персоналу, так і кінцевих користувачів, затримки з оновленням ОС і так далі.

Друга категорія технологій ґрунтується на аналізі вмісту пам'яті та жорстких накопичувачів. Найчастіше цю роль виконує антивірусний software різної спрямованості та методології (пошук по сигнатурах, пошук по евристикам). Цей метод також є досить повільним, особливо з урахуванням зростання обсягів інформації, що зберігається і передається, і обсягів самого software.

Одним з основних засобів як первинної, так і вторинної діагностики вторгнення є журнали подій (логи) програмного забезпечення та системні журнали (логи операційної системи) – набори діагностичних повідомлень, які генеруються штатно або нештатно функціонуючим програмним забезпеченням і можуть бути ефективно використані для виявлення факту вторгнення – з прийняттям подальших дій.

Логи часто є єдиним способом для роботи з розкриття інцидентів, їх аналіз може проводитися різними способами – як ручним, так і автоматизованим – шляхом підбору сигнатур та евристичних правил, однак застосовуваних не до аналізу бінарних файлів, а до аналізу повідомлень у логах.

Логи є цінним матеріалом для аналізу характеру атак, поведінки сторони, що атакує, та інформування у разі прориву захисту. Робота з логами та системними журналами є важливим компонентом безпеки системи, тому, якщо розгортається інтегрована система безпеки, до неї в обов'язковому порядку включається SIEM – Security Information and Event Management, який раніше являв собою поєднання SIM (Security Information Management) – системи управління інформацією про безпеку, та SEM (Security Event Management) – системи управління подіями безпеки.

SIEM забезпечує аналіз у реальному часі подій безпеки (тригерів), що надходять з різних пристроїв та додатків, і дозволяє завчасно реагувати на можливі загрози. Сучасні SIEM-системи можуть мати

досить складні та розгалужені архітектури та включати безліч різних джерел інформації. Однак, якщо серед великих корпорацій використання SIEM широко поширене, то для веб-серверів середнього рівня характерна, по-перше, відсутність будь-якої інтегрованої системи безпеки, по-друге, нехтування збором, захистом та обробкою логів.

Таким чином, аналіз системних журналів (логів) є одним з найбільш оперативних і доступних методів виявлення вторгнення malware. Логи є цінним матеріалом для аналізу характеру атак, поведінки атакуючої сторони та інформування у разі порушення захисту [25].

1.3.5. Журнали подій та виявлення вторгнень

Деякий час захист системних журналів не був пріоритетом при розробці систем безпеки – особливо серед системних адміністраторів середньої та нижчої ланки ICN, які часто не володіють ресурсами та знаннями для організації розвиненої інфраструктури безпеки, а іноді й поєднують роботу із забезпечення кібербезпеки з будь-якими іншими службовими обов'язками. Додатковою перешкодою була відсутність єдиних стандартів, тактики та політики захисту системних журналів. Впровадження комплексної інтегрованої системи, як правило, представляє складність і часто не є пріоритетом при розробці та експлуатації ICN. Відповідно, найчастіше експлуатація ICN поза великими корпораціями зводиться до набору hardening практик і «корисних порад», які далеко не завжди виконуються системними адміністраторами. Це вимагає організації інфраструктури, яка з одного боку забезпечувала б збирання, аналіз та зберігання системних журналів, з іншого боку дозволяла запобігти фальсифікації показань системних журналів і – з третьої сторони не була б ресурсозатратною як у розгортанні, так і в експлуатації. Останнє важливо, оскільки на будь-яких системах рівня нижче корпоративного, складність та витратність в експлуатації часто призводить до того, що рішення не запроваджується взагалі.

Адміністратори рідко беруть на себе працю організувати резервне копіювання, обробку та аналіз логів. При цьому резервна копія, навіть якщо вона реалізована, рідко перевіряється на цілісність та консистентність – чим цілком можна скористатися для маскування зламу системи шляхом підміни або спотворення логів, які можуть його виявити.

Причин такого нехтування декілька:

– не всі виділяють логи та системні журнали як значущий компонент безпеки системи (часто обмежуються встановленням брандмауера та антивірусу);

– логи займають великий обсяг (наприклад лог не найбільш навантаженого сервера за 8 днів становить понад 400 Мб);

– робота з логами вимагає витрат на організацію зовнішньої інфраструктури, при цьому, залежно від обсягу логів, така інфраструктура може виявитися досить дорогою та недостатньо ефективною;

– не склалося єдиного стандарту або практики роботи з логами;

– стандарт на повідомлення логів (Sigma – Generic Signature Format for SIEM Systems) ще не отримав загального визнання.

При цьому формат логів та їхня обробка часто залишається на розсуд організатора сервера – і йому не приділяється належної уваги.

Максимум захисту, який може здійснюватися в такому випадку, це:

– резервне копіювання логів;

– аналіз логів автоматичним аналізатором;

– аналіз логів за власними правилами.

Цього явно недостатньо – особливо з урахуванням зростання числа вторгнень та статистики щодо їхнього виявлення – навіть у великих корпораціях, що використовують SIEM. Показовою є поява нових типів шкідливого ПЗ, яке вже націлене на роботу з логами – хоча і досить примітивним чином.

На тлі всього вищезазначеного, логів як засіб виявлення malware поступово перетворюється на мету для проєктувальників malware, які чудово обізнані про ці технології і намагаються їх обійти.

У 2013 році корпорація MITRE анонсувала базу знань ATT&CK (Adversarial Tactics, Techniques & Common Knowledge – Тактики, техніки та загальновідомі знання про зловмисників) як спосіб опису та категоризації поведінки зловмисників, заснований на аналізі реальних атак. MITRE ATT&CK є структурованим списком відомих поведінок зловмисників, розділених на тактики та методи, і виражених у вигляді таблиць (матриць). Матриці для різних ситуацій та типів зловмисників публікуються на сайті MITRE. Також класифікація доступна в машиночитаних форматах STIX / TAXII. Оскільки

цей список дає комплексне уявлення про поведінку зловмисників під час зламу мереж, він вкрай корисний для різних захисних заходів, моніторингу, навчання та інших застосувань.

Відповідно до класифікації MITRE [33], атаки, пов'язані з модифікацією логів (Indicator Removal on Host: Clear Linux or Mac System Logs), належать до метакласу з ID T1070, який включає наступні техніки:

- T1070.001 Clear Windows Event Logs;
- T1070.002 Clear Linux or Mac System Logs;
- T1070.003 Clear Command History;
- T1070.004 File Deletion;
- T1070.005 Network Share Connection Removal;
- T1070.006 Timestomp.

Зокрема, туди входять техніки T1070.001 (Очищення журналів подій Windows) і техніка T1070.002 (Очищення системних журналів Linux або Mac). Ці дві техніки стосуються розглядуваного нами виду атак на систему повідомлень.

У 2020 році, в доповіді [34] ми припустили появу malware, яка маскуватиме факт вторгнення в систему шляхом заміни записів у системних журналах і запропонували технологію захисту, засновану на зв'язаних списках.

На той момент у базі даних MITRE було зареєстровано 2 різновиди malware, що використовують техніку T1070.001 (Очищення журналів подій Windows) та 1 різновид, що використовують техніку T1070.002 (Очищення системних журналів Linux або Mac).

У 2022-му році ця цифра склала 17 типів для T1070.001 і 3 типу для T1070.002.

На даний момент зареєстровано 28 типів, що використовують техніку T1070.001 та 4 різновиди, що використовують техніку T1070.002. Це тільки початок – зараз основною технологією маскування malware є просте видалення записів із системного журналу. Проте, з розвитком систем виявлення очікується появи malware, яке навчиться видаляти чи підробляти записи у системному журналі вибірково те щоб не видавати факт вторгнення відсутністю записів взагалі.

Розглянемо докладніше двох представників шкідливого ПЗ, які використовують техніку T1070.002.

1.3.6. Приклади шкідливого програмного забезпечення, орієнтованого на маскування дій та маніпуляції з логами

Перший з прикладів – це Proton. Він з'явився у 2017 році та використовував новий на той момент вектор зараження – коли зловмисник отримував доступ до реального сайту виробника програмного забезпечення та «троянізував» його додатки. Таким чином, зараження відбувалося під час встановлення цілком законного програмного забезпечення – при цьому не спрацьовував захисний периметр системи.

Показово, що поширення через злам сайту продемонструвало високий KPI, і програма на самому початку свого існування продавалася на чорному ринку за високими цінами: 100 BTC за необмежену кількість машин для зараження, потім 40 BTC та 2 BTC за одиничний комп'ютер, що навіть за мірками 2017 року становило високі суми порядку десятків та сотень тисяч доларів – і дає уявлення про попит на подібне ПЗ.

Троян пропонував широкий спектр дій на комп'ютері користувача – збір конфіденційної інформації, скріншоти робочого столу тощо.

Стирання логів для маскування дій троянця здійснювалося за допомогою команди, що видаляла системні логи і таким чином приховувала сліди дій на комп'ютері користувача:

```
sudo -k; echo '%@' | sudo -S rm -rf /var/log/* /Library/Logs/* && echo success;
```

Другий представник – криптомайнер Rocke – є характерним зразком нового покоління шкідливого програмного забезпечення.

Про групу зловмисників Rocke вперше стало відомо у 2018 році, коли в «медову пастку» (honeypot) потрапили перші зразки криптомайнера. Перші зразки були написані на Python, потім мову змінили на Go.

Для первинного впровадження Rocke використовує вразливості у веб-фреймворках Apache Struts 2, Oracle WebLogic та Adobe ColdFusion. Наприклад, у випадку Oracle використовується вразливість CVE-2017-10271, яка дозволяє отримати контроль над сервером та завантажити туди шкідливе програмне забезпечення.

Після впровадження запускається складний процес захоплення системи, що використовує мережеві сервіси paste.bin

(для завантаження додаткових шкідливих скриптів) та ідентифікатор ident.me для організації атак та розповсюдження на інші суміжні із захопленою системи.

Шкідлива програма виконує цілий набір дій:

- прописує себе в `crontab` так, щоб перезапускатися при кожному рестарті системи;
- ховає себе у списку процесів Linux за допомогою інструменту `libprocessehider`;
- завантажує запакований UPX майнер зі стороннього сайту. При цьому заголовок UPX спеціально модифікований, щоб зламати розпакувальник UPX. Замість "UPX!" рядок був замінений на "LSD!". Для розпакування зразків за допомогою розпакувальника, наданого командою UPX, необхідно вручну змінювати заголовки;
- зупиняє та деінсталює захисне програмне забезпечення Alibaba Cloud та Tencent Cloud – Alibaba Threat Detection Service та Tencent Cloud Host Security.

Цікава особливість шкідливої програми полягає в тому, що вона прагне позбутися інших криптомайнерів, оскільки за системні ресурси йде жорстка конкуренція. Для цього вона:

- модифікує дату та час шкідливих файлів так, щоб уникнути виявлення під час пошуку за метаданими;
- знаходить та знищує інші криптомайнери;
- змінює таблиці маршрутизації так, щоб заблокувати роботу інших криптомайнерів.

Нарешті, криптомайнер видаляє логи, надсилаючи нульове значення в наступні файли:

```
echo 0>/var/spool/mail/root
echo 0>/var/log/wtmp
echo 0>/var/log/secure
echo 0>/var/log/cron
```

Перші версії криптомайнера були написані мовою Python, згодом команда перейшла на використання дропера, написаного на Go. Системи виявлення на 2019 рік практично не помічали цей дропер.

Низький рівень виявлення шкідливого ПЗ у поєднанні з методами, що запобігають появі шкідливих процесів Rocke у запущених

процесах на машинах-жертвах, дозволив шкідливій програмі успішно розповсюджуватися протягом кількох тижнів.

Для перелічених вище випадків характерні наступні спільні моменти:

– зловмисники вже почали практикувати включення до шкідливого ПЗ інструментів маскування присутності шкідливих програм у системі, зокрема шляхом модифікації логів, які добре доповнюють інші інструменти маскування (timestomping, маскування процесів тощо);

– наявність навіть найпримітивнішої атаки – простого видалення логів – сильно знижує помітність наявності шкідливого ПЗ у системі, ускладнює виявлення та діагностику шкідливого ПЗ та дозволяє руткітам тривалий час функціонувати без виявлення – що, своєю чергою, збільшує масштаби зараження.

При цьому робота з логами та журналами системних подій перебуває на початковій стадії – як було зазначено вище, атака шляхом видалення логів спрацьовує лише за відсутності належної інфраструктури роботи з логами та журналами подій та низької грамотності системного адміністратора. За наявності елементарних засобів захисту, перелічених вище, віддалені логи будуть відновлені з резервної копії, а сам факт відсутності записів у системному журналі вказуватиме на наявність зовнішнього вторгнення та послужить приводом для вжиття всіх необхідних заходів щодо захисту системи.

Як аналіз логів з метою визначення вторгнення в систему, так і протидія йому все ще перебувають на початковому етапі розвитку. Нам видається, що прогрес у засобах виявлення потягне за собою появу шкідливого програмного забезпечення, яке матиме розвинений інструментарій роботи з логами та від простого знищення перейде до підробки логів з метою маскування своєї присутності в системі.

1.3.7. Забезпечення цілісності логів: від зв'язкових Списків до Verkle Trees

У роботах [35] було запропоновано таке рішення, що ґрунтувалося на технології зв'язкових списків. У цьому рішенні додатковий контроль за цілісністю системних журналів зводився до того, що в системний

журнал через деякі проміжки часу додавали спеціальне повідомлення, що представляє собою хеш-суму з повідомлень, що потрапили в системний журнал, включаючи передостаннє контрольне повідомлення. Таким чином, журнал з одного боку верифікувався наявністю ланцюжка контрольних сум, з іншого така схема не вимагала великих обчислювальних витрат, інфраструктури та гнучко налаштовувалась, оскільки інтервал можна було задавати як тимчасовим, так і за кількістю повідомлень у системному журналі. Така система добре показала себе на практиці на серверних системах середнього та низького рівня, проте, у зв'язку з широким поширенням хмарних систем, все гострішою стає необхідність в аналогічних заходах захисту, але вже дозволяють контролювати розподілені системи – де реалізація простої схеми зв'язкових списків не є можливою технічно.

У роботі [36] було запропоновано розвиток системи шляхом організації перехресних перевірок цілісності на базі технології дерев хешів, найвідомішою реалізацією яких є Merkle Tree. Використання Merkle tree виглядає наступним чином – на основі хешів вихідних записів вибудовується дерево хешів, в якому хеш верифікуються попарно, утворюючи собою ієрархію – дерево хеш, яке зі зростанням бази даних росте пропорційно `O(log N)`. Використання даної схеми роботи дозволить з одного боку зберегти всі переваги системи, заснованої на хеш-сумах (невимогливість до ресурсів, відсутність необхідності у побудові та обслуговуванні складної інфраструктури), з іншого, шляхом незначного збільшення надмірності дозволило отримати такі переваги, як можливість роботи з розподіленими ресурсами, що покращить загальну безпеку системи.

Подальшим розвитком ідеї стало використання Verkle Trees [37], в якому хеш-дерево скорочено до кількох рівнів, що серйозно впливає на швидкість та простоту логів інфраструктури перевірки. Крім інших переваг, Verkle Tree дозволяє гнучко використовувати алгоритми різної ефективності та роботи – від звичайного хешування до схеми поліноміального зобов'язання KZG [38]. В [36] використовувалася криптографічна хеш-функція SHA-256, щоб спростити реалізацію Verkle Tree у різних контекстах використання завдяки наявності інструментальної бази у вигляді бібліотек, що добре зарекомендували себе.

1.4. Адаптація систем безпеки до несприятливих умов функціонування ICN та ICS

Як було сказано вище, централізація збору системних журналів в даний час додатково утруднюється через все збоїв у мережі (блекаути, кібератаки, бойові дії і т. д.), що частішають, при яких мережа може частково або повністю втрачати зв'язність і розпадатися на незалежні сегменти з подальшим відновленням. Існуючі системи SIEM передбачають можливість практично миттєвого доступу до ключової інформації обсягом всього підконтрольного сегменту ICN. Короткочасна або тривала втрата взаємодії з підсистемами ICN може призвести до розладу та плутанини у існуючих системах контролю цілісності балок.

Отже, існує необхідність в інфраструктурі збору та верифікації системних журналів у рамках розподіленої ICN, яка не вимагала б складної інфраструктури та витрат на впровадження та дозволяла б функціонувати ICN в умовах часткової доступності її окремих «острівів» – напівавтономна або цілком автономна на деякі періоди часу. Існуючі рішення зазвичай зводяться до резервування та дублікації в тому чи іншому вигляді системних журналів по всій системі централізованим або централізованим чином [39; 40], що вимагає великих вкладень (наприклад, фактично подвоюється або потроюється необхідна ємність сховищ інформації) не забезпечує оперативного виявлення порушень цілісності системних журналів, оскільки у межах такої системи її можна виявити лише шляхом послідовної звірки записів оригіналу та копії. При цьому слід враховувати, що кібератаці або несприятливому впливу може бути схильний до будь-якого з рівнів системи.

У роботі пропонується схема асинхронної взаємодії систем верифікації логів. В рамках цієї системи низові ланки ICN працюють згідно зі схемою, наведеною в [35], де кожен журнал верифікується шляхом технології зв'язкових списків. Система розширюється за рахунок додавання до кожної контрольної суми метаданих: дати та внутрішнього часу обчислення дайджесту, кількості завірених повідомлень, а також ідентифікатора вузла ICN. Це є природним розширенням системи ведення логів і часто не вимагає перегляду схеми,

оскільки такі метадані додаються до запису автоматично самим програмним забезпеченням для ведення логів. Однак, замість регулярного збору інформації по системі, шляхом запитів від вищих вузлів ієрархії ICN, після кожного обчислення контрольної суми, низовий елемент виконує в термінології запропонованої системи дію PUSH, самостійно відправляючи запит «квитанцію» на рівень вище. Це може бути реалізовано як з підтвердженням отримання квитанції вищим вузлом, так і без неї. У свою чергу, кожен вищий вузол збирає отримані «квитанції». Після досягнення певної кількості квитанцій, він аналогічним чином обчислює свій дайджест, додає до нього метадані та надсилає отриману квитанцію ще на один рівень вище. Таким чином вибудовується ієрархічна система, подібна до Merkle Tree, яка дозволяє працювати асинхронно і без безперервного доступу до всіх мережевих сегментів ICN. У разі збою зв'язку повідомлення з конкретного вузла будуть просто надіслані пізніше.

При необхідності аудиту системи, вищестоящий вузол відправляється запит PULL, який нижчестоящі вузли дублюють ієрархії вниз. Після чого вузли, до яких прийшов PULL-запит знову виконують процедуру PUSH за всіма запитаними даними, а вузол, що надіслав запит PULL звіряє отримані повторно дайджести з централізовано. Така схема дозволяє з одного боку працювати в умовах тимчасових втрат зв'язку та зв'язності, з іншого боку за рахунок асинхронності дозволяє рівномірно розподіляти навантаження в ICN не перевантажуючи вузли-збирачі інформації великим обсягом обчислень.

1.5. Висновки

Аналіз поданих матеріалів дає змогу сформулювати низку ключових висновків щодо стану, вразливості, живучості та безпеки сучасних інформаційно-комунікаційних систем (ICN), зокрема веб-серверів.

Сучасні ICN, попри їхню складність та масштаб, апріорі вразливі до численних кібератак та несприятливих впливів. Тенденції до централізації та ієрархізації, часто без належного архітектурного планування, призвели до втрати гнучкості та здатності ефективно протистояти загрозам. У поточних умовах основним завданням

кібербезпеки ICN стає не стільки запобігання атакам (що є майже неможливим), скільки мінімізація часу відновлення функціональності після інциденту. Це підтверджується досвідом великомасштабних збоїв, таких як інцидент з «Київстар» у грудні 2023 року, коли повне відновлення зайняло близько семи днів. Веб-сервери є ключовими, детермінуючими компонентами ICN, що забезпечують безперебійне функціонування життєво важливих сервісів (від фінансових до комунальних). Їхній вихід з ладу призводить до каскадних відмов та мультиплікації збитків по всій мережі, як це було продемонстровано під час збоїв мобільних мереж, що вплинули на банківський сектор та комунальні служби. Події останніх років, включаючи блекаути та атаки на критичну інфраструктуру, підтвердили критичну важливість концепції живучості мережі. Це вимагає комплексного прогнозування та сценарного моделювання можливих збоїв та каскадного виходу з ладу систем, а також розробки моделей відновлення, що враховують вторинні ефекти та ризики.

Кількість, складність та інтелектуальність кібератак постійно зростають, зловмисники вдосконалюють свої техніки, тактики та стратегії. Руткіти та програми-вимагачі (наприклад, Stuxnet, Triton, WannaCry, Petya, Snake) еволюціонують від «традиційного» вандалізму до прихованої діяльності, спрямованої на збір конфіденційної інформації та організацію управління атаками. Традиційні захисні механізми, такі як «повітряний зазор», виявляються недостатніми для протидії сучасним загрозам, що підтверджено реальними інцидентами. Попри зусилля, середній час виявлення успішної кібератаки залишається надзвичайно великим (від 50–70 днів у 2015 році до 280 днів для ідентифікації витоків у 2020 році). Це дозволяє шкідливому програмному забезпеченню функціонувати непоміченим протягом тривалого часу, збільшуючи масштаби ураження та збитків. Класичні методи виявлення, такі як сигнатурний та евристичний аналіз бінарних файлів, хоч і цінні, мають значні недоліки: трудомісткість оновлення баз даних, вразливість до поліморфного та самомодифікованого ПЗ, а також високий рівень хибних спрацьовувань для евристик. Аналіз мережевої та системної активності також ускладнюється через використання шифрування та фрагментації мереж.

Завдяки своїй відкритості, постійному підключенню до зовнішніх мереж, повсюдному поширенню та ролі як інтегруючого елемента, веб-сервери стали основною мішенню для атак ботнетів та інших шкідливих програм. Їхня складна, багатокомпонентна архітектура та часто низька кваліфікація персоналу створюють численні вразливості. Журнали подій (логи) є основним засобом діагностики вторгнень та розкриття інцидентів. Проте, їхній величезний обсяг, відсутність стандартизації та недостатня увага до їхнього збору, захисту та обробки (особливо на середніх веб-серверах) є серйозною проблемою. Існуючі системи SIEM, попри свою ефективність, часто є надмірно складними та дорогими для широкого впровадження. З огляду на неминучість інцидентів, раціональна організація систем резервного копіювання (бекапів) є найважливішим елементом мінімізації часу відновлення працездатності ICN. Це стосується не лише даних, а й усієї інфраструктури веб-сервера (ОС, БД, ПЗ тощо). Для забезпечення стійкості та надійності систем резервного копіювання необхідно дотримуватися перевірених принципів: фізичне розділення робочої системи та сховища бекапів, шифрування резервних копій, обов'язкове тестове відновлення, автоматизація процесів резервування з документуванням у журналах, а також організація ротації кількох копій для ефективного використання ресурсів та збереження інформації.

Таким чином, для забезпечення живучості та стійкості сучасних ICN необхідний інтегрований, багатофакторний підхід. Він має включати не лише вдосконалення методів виявлення та протидії кіберзагрозам, а й розробку надійних стратегій відновлення, що базуються на ефективному резервному копіюванні, врахуванні вторинних ефектів збоїв та постійному аналізі еволюції загроз. Запропонована система контролю та управління розподіленими системами виявлення вторгнень з використанням логів у несприятливих умовах використовує асинхронний режим роботи та дозволяє з одного боку зберігати та підтримувати цілісність логів в умовах несприятливих впливів та вражаючих факторів, з іншого боку за рахунок асинхронності дозволяє рівномірно розподіляти навантаження в ICN, не перевантажуючи вузли-збирачі інформації великим обсягом обчислень. Впровадження та розгортання системи на запропонованих

принципах вимагає менше ресурсів у порівнянні з існуючими SIEM-системами і може бути здійснене як самостійно, так і з інтеграцією у вже розгорнуту SIEM, якщо вона є. Запропоновані заходи дозволять значно збільшити безпеку, оперативність реагування на інциденти та можливість відновлення існуючих ICN.

Список використаних джерел

1. Bi W., MacAskill K., Schooling J. Old wine in new bottles? Understanding infrastructure resilience: Foundations, assessment, and limitations. *Transportation Research Part D: Transport and Environment*. Elsevier BV, 2023. Vol. 120. P. 103–793. URL: http://dx.doi.org/10.1016/j.trd.2023.103793
2. Бойко В., Василенко М., Слатвінська В. Моделювання живучості та відновлення інформаційно-комунікаційних мереж в умовах дії кібер-загроз. *Інформаційні технології та суспільство*. 2024. № 1 (12). P. 13–19. URL: https://journals.maup.com.ua/index.php/it/article/view/3143
3. Олена К. Збій у роботі мережі Київстар – оновлення та рекомендації для абонентів. URL: https://biz.nv.ua/ukr/tech/zbiy-u-robo ti-merezhi-kijivstar-onovlennya-ta-rekomendaciji-dlya-abonentiv-50375659html (дата звернення: 2024–11–06).
4. Олена К. У Київстарі розповіли про наслідки кібератаки на 3,6 млрд грн. URL: https://biz.nv.ua/ukr/markets/u-kijivstari-rozpovili-pro-naslidki-kiberataki-na-3-6-mlrd-grn-50419968.html (дата звернення: 2024–11–06).
5. Інформація про Vodafone – Forbes Ukraine. URL: https://forbes.ua/profile/vodafone-251 (дата звернення: 2023–12–24).
6. Інформація про Lifecell – Forbes Ukraine. URL: https://forbes.ua/profile/lifecell-574 (дата звернення: 2023–12–24).
7. Інформація про Kyivstar – Forbes Ukraine. URL: https://forbes.ua/profile/kiivstar-244 (дата звернення: 2023–12–24).
8. Через несправність мобільного оператора Київстар відключення вуличного освітлення відбуваються в ручному режимі – Львівська міська рада. URL: https://city-adm.lviv.ua/news/city/housing-and-utilities/299531-cherez-nespravnistiu-mobilnoho-operatora-kyivstar-vid kliuchennia-vulychnoho-osvitlennia-vidbuvaiutsia-v-ruchnomu-rezhymi (дата звернення: 2023–12–24).

9. Boyko V., Vasilenko M., Slatvinska V. Survivability and sustainability of smart city information system components. *Municipal economy of cities*. O. M. Beketov National University of Urban Economy in Kharkiv, 2021. Vol. 6 (166). P. 20–27. URL: https://khg.kname.edu.ua/index.php/khg/article/view/5861

10. Barzashka I. Are Cyber-Weapons Effective? Assessing Stuxnet's Impact on the Iranian Enrichment Programme. *The RUSI Journal.* Taylor & Francis, 2013. Vol. 158(2). P. 48–56. URL: https://www.tandfonline.com/doi/abs/10.1080/03071847.2013.787735

11. Di Pinto A., Dragoni Y., Carcano A. TRITON: The first ICS cyber attack on safety instrument systems. *Proc. Black Hat USA*. 2018. P. 1–26.

12. Lee R. TRISIS Malware: Analysis of Safety System Targeted Malware. Dragos Inc. 2017. URL: https://www.dragos.com/wp-content/uploads/TRISIS-01.pdf

13. Case D. U. Analysis of the cyber attack on the Ukrainian power grid. *Electricity Information Sharing and Analysis Center (E-ISAC).* 2016. Vol. 388. URL: https://ics.sans.org/media/E-ISAC_SANS_Ukraine_DUC_5.pdf

14. Slowik J. CRASHOVERRIDE: Reassessing the 2016 Ukraine electric power event as a protection-focused attack. Dragos, Washington, DC, USA, Tech. Rep., Aug. 2019. URL: https://www.dragos.com/wp-content/uploads/CRASHOVERRIDE.pdf

15. Fayi S. Y. A. What Petya / Not Petya ransomware is and what its remidiations are. Information Technology – New Generations. Springer, 2018. P. 93–100.

16. Slowik J. Evolution of ICS Attacks and the Prospects for Future Disruptive Events. Accessed: Feb, 2020. URL:https://www.dragos.com/wp-content/uploads/Evolution-of-ICS-Attacks-and-the-Prospects-for-Future-Disruptive-Events-Joseph-Slowik-1.pdf

17. Branquinho M. A. Ransomware in industrial control systems. what comes after wannacry and petya global attacks? *WIT Transactions on The Built Environment*. WIT Press, 2018. Vol. 174. P. 329–334.

18. Ransomware Against the Machine: How Adversaries are Learning to Disrupt. FireEye. 2020. URL: https://www.fireeye.com/blog/threat-research/2020/02/ransomware-against-machine-learning-to-disrupt-industrial-production.html

19. Mandiant. Mandiant Security Effectiveness Report. *FireEye.* 2020. P. 1–22. URL: https://www.fireeye.com/current-threats/annual-threat-report/security-effectiveness-report.html

20. *Бойко В. Д.* Моделювання кібернетичної складової живучості складних технічних систем. *Інформаційні управляючі системи та технології* : матеріали Міжнародної науково-практичної конференції (ІУСТ-Одеса-2014), 23–25 вересня 2014 року. Одеса, 2014. С. 239–241.

21. *Rausand M., Høyland A.* System Reliability Theory: Models, Statistical Methods, and Applications, Second Edition / 2nd ed. Wiley-Interscience, 2003. P. 636. URL: http://gen.lib.rus.ec/book/index.php?md5=5F7366820A70CEB471BD9BC7A29D4411

22. Cimellaro G. P., Reinhorn A. M., Bruneau M. Framework for analytical quantification of disaster resilience. Engineering Structures. Elsevier BV, 2010. Vol. 32 (11). P. 3639–3649. URL: http://dx.doi.org/10.1016/j.engstruct.2010.08.008

23. Sobhaninia S., Buckman S. T. Revisiting and adapting the Kates-Pijawka disaster recovery model: A reconfigured emphasis on anticipation, equity, and resilience. *International Journal of Disaster Risk Reduction. Elsevier BV.* 2022. Vol. 69. P. 102–738. URL: http://dx.doi.org/10.1016/j.ijdrr.2021.102738

24. Silva S. S. C., Silva R. M. P., Pinto R. C. G., Salles R. M. Botnets: A survey. Computer Networks. *Elsevier BV.* 2013. Vol. 57 (2). P. 378–403. URL: http://dx.doi.org/10.1016/j.comnet.2012.07.021

25. Barford P., Yegneswaran V. An Inside Look at Botnets. Advances in Information Security. *Springer US.* 2007. P. 171–191. URL: https://doi.org/10.1007%2F978-0-387-44599-1_8

26. *Blunden B.* The Rootkit arsenal: Escape and evasion in the dark corners of the system. Jones & Bartlett Publishers, 2012. P. 783.

27. Harley D., Lee A. The root of all evil? – Rootkits revealed. Online, 2007. P. 1–17.

28. Lee K., Lim J. The Reality and Response of Cyber Threats to Critical Infrastructure: A Case Study of the Cyber-terror Attack on the Korea Hydro & Nuclear Power Co., Ltd. KSII Transactions on Internet & Information Systems. 2016. Vol. 10 (2).

29. Johnson J. Average number of days to resolve a cyber attack on companies in the United States as of August 2015, by attack type.

URL: https://www.statista.com/statistics/193463/average-days-to-resolve-a-cyber-attack-in-us-companies-by-attack/ (дата звернення: 2022–06–10).

30. IBM. Cost of a Data Breach Report 2020. URL: https://www.ibm.com/security/digital-assets/cost-data-breach-report/#/ru (дата звернення: 2022–06–10).

31. Cinque M., Cotroneo D., Pecchia A. Challenges and Directions in Security Information and Event Management (SIEM) / 2018 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW). IEEE, 2018. P. 95–99. URL: https://doi.org/10.1109%2Fissrew.2018.00-24

32. González-Granadillo G., González-Zarzosa S., Diaz R. Security Information and Event Management (SIEM): Analysis, Trends, and Usage in Critical Infrastructures. Sensors. MDPI AG, 2021. Vol. 21 (14). P. 1–28. URL: https://doi.org/10.3390%2Fs21144759

33. Indicator Removal on Host: Clear Linux or Mac System Logs, Sub-technique T1070.002 – Enterprise MITRE ATT&CK. URL: https://attack.mitre.org/techniques/T1070/002 (дата звернення: 2022–06–10).

34. Бойко В., Василенко М. Система виявлення вторгненнь з використанням технології зв'язаних списків *Контроль і управління в складних системах (КУСС-2020)* : матеріали XV міжнародної конференції, м. Вінниця, 8–10 жовтня 2020 року, ВНТУ, 2020. С. 265–266. URL: http://ir.lib.vntu.edu.ua//handle/123456789/30577

35. Boyko V., Vasilenko M., Slatvinska V. Linked list systems for system logs protection from cyberattacks. *Information Technologies in Education, Science and Technology (ITEST-2022).* June 23–25, 2022 Cherkasy. 2022. P. 81–82. URL: https://er.chdtu.edu.ua/bitstream/ChSTU/4121/1/Збірник_тез_ІТОНТ-2022_макет_23_06.pdf#page=81

36. Boyko V., Vasilenko M., Slatvinska V. Linked List Systems for System Logs Protection from Cyberattacks. Information Technology for Education, Science, and Technics. ed. by Faure E., Danchenko O., Bondarenko M., Tryus Y., Bazilo C., Zaspa G. Springer Nature Switzerland, 2023. P. 224–234. URL: https://doi.org/10.1007%2F978-3-031-35467-0_15

37. Kuszmaul J. Verkle Trees. 2019. P. 11. URL: https://api.semanticscholar.org/CorpusID:218475793

38. Tas E. N., Boneh D. Vector Commitments with Efficient Updates. arXiv, 2023. P. 46. URL: https://arxiv.org/abs/2307.04085

39. Aßmuth A., Duncan R., Liebl S., Söllner M. A secure and privacy-friendly logging scheme. Cloud Computing 2021 / під ред. Bob Duncan, Yong Woo Lee, Manuela Popescu. International Academy, Research; Industry Association (IARIA), 2021. P. 8–12. URL: https://www.iaria.org/conferences2021/CLOUDCOMPUTING21.html.

40. Hangxia Z., Peng Z., Yong Y. Web log system of automatic backup and remote analysis. *International Conference on Computer Application and System Modeling (ICCASM 2010).* IEEE, 2010. P. 469–472. URL: https://doi.org/10.1109%2Ficcasm.2010.5620567.