УДК 004.056:[004.73+004.451
DOI: 10.31673/2412-9070.2024.065177
О. Ю. КОНОВАЛОВ, канд. техн. наук, доцент;
ORCID: 0000-0002-3772-3654
Ю. Є. ДОБРИШИН, канд. техн. наук, доцент;
ORCID: 0000-0003-2473-9507
С. М. СИДОРЕНКО, старший викладач,
ORCID: 0009-0003-1185-1505
Національна академія Служби безпеки України, Київ

МОДЕЛЮВАННЯ ДОСТУПУ ДО ВІДДАЛЕНОЇ МЕРЕЖЕВОЇ СИСТЕМИ З IDS SNORT, З ВИКОРИСТАННЯМ МУЛЬТИПЛЕКСОРА SSL / SSH (SSL/SSH-ПРОКСІ)

У статті розглянуто питання використання мультиплексора SSL/SSH (SSL/SSH-проксі) для організації доступу до віддаленої мережевої системи з IDS SNORT при підготовці дисциплін з кібербезпеки. Враховуючи можливості мультиплексора протоколів SSL/SSH (SSL/SSH-проксі), розглянуто принципи та механізми організації доступу до віддаленої операційної системи Linux Ubuntu, з розгорнутою на ній IDS системою Snort, проведений аналіз роботи і режимів налаштування IDS Snort для контролю такого типу підключення з використанням різних типів протоколів. Наведені приклади налаштування конфігураційних файлів як для мультиплексора SSL/SSH (SSL/SSH-проксі), так і приклади налаштування IDS системи Snort, виконана перевірка роботи обох програмних продуктів при використанні різних протоколів і портів для організації віддаленого доступу до операційної системи Linux Ubuntu.

З метою моделювання доступу до віддаленої системи, перевірка виконувалась з використанням засобів IDS SNORT, фізичного обладнання на базі операційної системи Windows 10, гіпервізора першого типу і віртуальних машин з операційними системами Linux Ubuntu.

За результатами досліджень виявлено, що використання мультиплексора SSL/SSH (SSL/SSH-проксі) разом із Snort дає можливість аналізувати зашифрований трафік, вирішуючи ключові проблеми, пов'язані з безпекою, зокрема виявлення витоків даних, шкідливого ПЗ, підозрілих підключень і аномальної активності. Це значно розширює можливості проведення аудиту комп'ютерних мереж і обладнання, покращує здатність систем до детектування і виявлення вторгнень, дозволяє виконувати функції моніторингу в умовах сучасних загроз, де велика частина трафіку шифрується для захисту або приховування шкідливої активності.

По результатам роботи наведені переваги використання віртуалізації для розгортання тестового мережевого середовища на базі мультиплексора SSL/SSH.

Для дослідження роботи мультиплексора протоколів SSL/SSH і IDS Snort, в якості організації лабораторного практикума, була вибрана лабораторна робота з моделювання розгортання мультиплексору протоколів SSL/SSH і аналізом мережевого трафіку засобами IDS Snort.

Ключові слова: мультиплексор протоколів SSL/SSH, IDS Snort, віддалений доступ, аналіз мережевого трафіку.

Вступ

Постановка проблеми. Використання мультиплексора протоколів SSL/SSH (sslh**)** разом із аналізом мережевого трафіку засобами системи виявлення вторгнень (IDS) Snort може допомогти вирішити частину проблем у сфері мережевої безпеки. Зашифрований трафік (через SSL/TLS або SSH) важко аналізувати через те, що дані не доступні для прямого перегляду і аналізу інструментами IDS, такими як Snort, тому що зашифрований трафік часто використовується для приховування витоків конфіденційної інформації, таких як номери кредитних карток, паролі або особисті дані.

Атакуючі можуть використовувати зашифрований трафік (SSH/SSL) для передачі шкідливого коду або команд до скомпрометованих машин, залишаючись непоміченими IDS. Шифрування може використовуватися для обходу засобів IDS, маскуючи трафік, що містить аномальну або підозрілу активність. Використання фальшивих або скомпрометованих сертифікатів SSL/TLS може дозволити зловмисникам виконувати атаки "людина посередині" або інші шкідливі дії. Окрім того, сучасні мережі можуть використовувати одночасно кілька протоколів (HTTP, HTTPS, SSH, FTP), що ускладнює моніторинг і аналіз трафіку за допомогою IDS. Застосування SSLH дозволяє вирішувати вказану проблему, шляхом мультиплексування кількох протоколів на одному порту, надаючи Snort змогу одночасно аналізувати на цьому порту трафік декількох типів для поглибленого аналізу.

Аналіз останніх досліджень. Snort дає можливість аналізувати трафік, вирішуючи ключові проблеми, пов'язані з безпекою, зокрема виявлення аномалій потоків даних, шкідливого ПЗ, підозрілих підключень і дій. Це значно покращує здатність системи IDS виконувати свої функції в умовах сучасних загроз, де велика частина трафіку шифрується для захисту або приховування шкідливої активності [1,2].

Для моделювання топології, необхідної для виконання роботи, будемо використовувати гіпервізор першого типу, до якого відносяться всі гіпервізори, що встановлюються на «голе залізо» так зване (bare metal), на якому у якості віртуальних машин однорангової мережі розгортаються дві операційні системи з Linux Ubuntu [2], а також використовується одна машина у локальній мережі з операційною системою Windows 10 на стаціонарному персональному компютері, з якого буде перевірятись робота налаштованої віртуальної локальної мережі, до якої будуть підключені віртуальні машини з операційними системами Linux Ubuntu: VM1 і VM2. При розгортанні віртуальних машин VM1 і VM2 потрібно налаштовувати мережеві карти у режимі бріджа (мосту), що дасть можливість ввести ці машини в мережу, до якої підключена машина з Windows 10. IP-адреси машини отримають від DHCP-сервера у локальній мережі, а доступ до Інтернету здійснюється через шлюз з адресою мережевого інтефейса 192.168.0.1/24. Якщо використовувати віртуальні машини, то потрібно розгорнути мультиплексор протоколів SSL/SSH і IDS Snort на одному хості, оскільки трафік з вхідного мережевого інтерфейса буде перехоплюватись (SSL/SSH-проксі) і перенаправлятись на інтерфейс, який буде прослуховувати IDS Snort. При наявності мережевої активності трафік буде аналізуватись програмним забезпеченням Snort відповідно до тих правил, які прописані в конфігураційних файлах вказаної IDS. При співпадінні правила і виявленої події, Snort виведе на екран повідомлення про цю подію і її назву. Оскільки моделювання потрібно виконувати у мережевому середовищі, потрібно також налаштувати віртуальну мережу для віртуальних машин згідно показаній на (рис. 1) топології. Якщо моделювання буде виконуватися на гіпервізорі другого типу, мережа може відрізнятись в деталях від показаної на рис. 1, так само як і IP-адресація, але розміщення на VM2 наступних модулів: мультиплексор протоколів (SSL/SSH-проксі) і IDS Snort є обовязковим для налаштування зазначеної мережі [3,4].

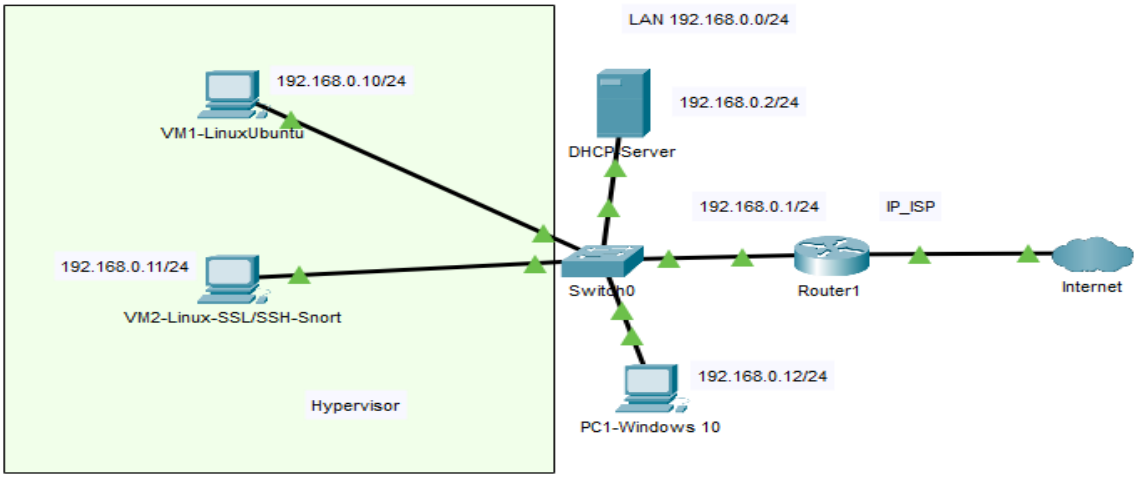

Рис. 1. Топологія мережі з SSLH-проксі і віддаленої мережевої системи з IDS SNORT

Постановка задачі. Проаналізувати методи використання мультиплексора протоколів SSL/SSH і IDS Snort як окремо так і в сумісництві. Дослідити можливості використання віртуального середовища запропонованої топології для організації лабораторного практикуму з кібербезпеки. Виконати моделювання мультиплексування протоколів на окремий порт і перевірити роботу Snort як засобу виявлення аномальної активності такого типу на прикладі лабораторної роботи.

Основна частина

SSL/SSH - це мультиплексор (SSLH-проксі), який дозволяє одночасно приймати кілька різних видів трафіку, наприклад, таких як HTTPS, SSH, та інші, на одному порту.

Основною метою SSLH-проксі є економія портів, оскільки зазвичай для кожного з протоколів потрібен окремий порт. SSLH-проксі працює як проксі-сервер і допомагає розподіляти трафік до відповідних сервісів на основі його характеристик, таких як структура пакету або поведінка при встановленні з'єднання [6,7]. SSLH-проксі дозволяє одночасно використовувати один порт для декількох служб, наприклад, для SSH і HTTPS на порту 443 або іншому. Це особливо корисно, коли необхідно обійти обмеження брандмауера або використовувати декілька протоколів через один порт з обмеженою кількістю доступних портів або можливостей конфігурації. SSLH-проксі не потребує значних ресурсів для роботи.

SSLH-проксі був розроблений з метою дозволити SSH-з'єднання через порт 443, який зазвичай використовується для HTTPS-трафіку, щоб обійти брандмауери. З часом програма отримала підтримку інших протоколів [6,7]. Алгоритм роботи SSLH виглядає наступним чином:

1. SSLH отримує вхідний трафік на зазначеному порту.
2. Далі здійснюється аналіз заголовків пакетів, щоб визначити, до якого протоколу належить трафік.
3. Після визначення, який саме протокол використовується, SSLH-проксі перенаправляє трафік на відповідний сервіс або порт.
4. Наприклад, якщо трафік використовує протокол HTTPS, то він перенаправляється на веб-сервер, а якщо SSH, то на SSH-сервер.

SSLH-проксі найчастіше використовується у середовищах, де потрібно одночасно обробляти декілька типів зашифрованого трафіку на одному порту, наприклад, у великих компаніях, дата-центрах, або для особистого користування адміністраторами мереж. SSLH-проксі сам по собі не забезпечує шифрування або захист даних, але працює з зашифрованими протоколами, такими як SSH або HTTPS. Використання SSLH практично не впливає на продуктивність системи, навантаження лягає на кінцеві сервіси (SSH, HTTPS) [8] [9].

Snort — це система виявлення вторгнень (IDS/IPS), яка дозволяє здійснювати моніторинг мережевого трафіку у реальному часі, аналізувати пакети і фільтрувати їх на основі певних правил. При тестуванні на проникнення Snort використовується під час симуляцій атак і тестування систем на вразливості [8,9]. На рис. 2 показаний приклад виявлення Snort сканування мережевого інтерфейсу сканером nmap

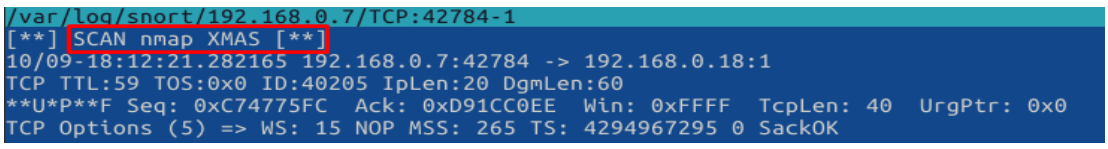

Рис. 2. Приклад виявлення Snort сканування мережевого інтерфейсу сканером nmap

Snort не шифрує трафік самостійно, але забезпечує детектування загроз у зашифрованому і незашифрованому трафіку за допомогою аналізу сигнатур та аномалій. Його безпека значною мірою залежить від своєчасного оновлення правил і конфігурації. Залежно від обсягу мережевого трафіку та кількості встановлених правил, Snort може впливати на швидкодію мережі. Але, не зважаючи на це, Snort залишається однією з найпопулярніших та надійних IDS/IPS систем у світі завдяки своїй простоті, ефективності та гнучкості у налаштуванні правил.

Інсталяція і конфігурування програмного забезпечення

Для дослідження використання SSLH-проксі і Snort IDS необхідно використовувати топологію, як показано на рис. 1, яка ілюструє запланований потік трафіку між SSLH-проксі і Snort IDS від комп'ютерів користувачів та веб-сервером і ssh-сервером, (рис. 3).

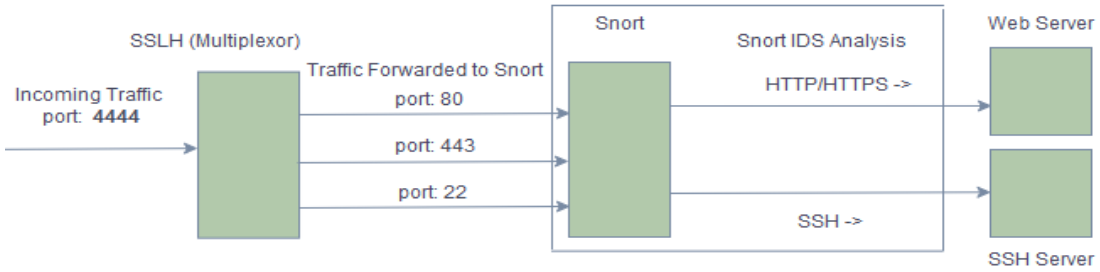

Рис. 3. Шлях трафіку у мережі з мультиплексуванням портів і Snort IDS

Для роботи на віртуальній машині VM2 встановлено SSL/SSH-проксі для дешифрування трафіку і перенаправлення відомостей щодо аналізу на відповідні порти Snort IDS. Налаштування SSLH-проксі і Snort IDS на віртуальної машині VM2 передбачає віддалено підключення з PC1 до віртуальної машини VM2 за допомогою протоколу SSH.

З метою встановлення SSLH-проксі і Snort IDS на віртуальну машину VM2 на базі операційної системи Linux Ubuntu 22, на зазначеному засобі необхідно спочатку виконати оновлення системи командами:

~$ sudo apt update | sudo apt upgrade

Після оновлення перевірити роботу SSLH-проксі (рис. 4) командою:

~$ sudo systemctl staus sslh

```
linaro@linaro-Standard-PC-i440FX-PIIX-1996:~$ sudo systemctl status sslh
[sudo] password for linaro:
● sslh.service - SSL/SSH multiplexer
     Loaded: loaded (/lib/systemd/system/sslh.service; enabled; vendor preset: enabled)
     Active: active (running) since Mon 2024-10-21 18:17:03 EEST; 1 day 22h ago
       Docs: man:sslh(8)
   Main PID: 26926 (sslh)
```

Рис.4. Перевірка роботи SSLH-проксі

У багатьох дистрибутивах операційної системи Linux програмний пакет SSLH-проксі є за замовчуванням. Якщо немає, необхідно виконати інсталяцію вказаного програмного пакета:

~$ sudo apt install sslh

Після цього встановити Snort та оновити його до останньої версії командами:

~$ sudo apt install snort

~$ sudo apt-get upgrade snort

Далі створити резервну копію конфігураційного файла snort.conf командою:

~$ sudo cp snort.conf snort.old.conf

Для подальшої роботи щодо налаштування програмного забезпечення Snort відкриваємо файл snort.conf наступною командою:

~$ sudo nano /etc/snort/snort.conf

Прописуємо адресу мережі, яку буде прослуховувати Snort (рис.5), для чого використовуємо відкриті доступні правила [4],[8] і вказуємо файл зі встановленими правилами local.rules (рис.6).

```
# /etc/snort/snort.debian.conf configuration file
#
ipvar HOME_NET 192.168.0.0/24
```

Рис. 5. Встановлення адреси мережі, яку буде прослуховувати Snort

```
# site specific rules
include $RULE_PATH/local.rules
#include $RULE_PATH/app-detect.rules
```

Рис. 6. Вигляд файлу зі встановленими правилами local.rules

Для подальшої роботи переходимо у каталог з правилами
cd /etc/snort/rules/
та відкриваємо для редагування файл local.rules
~$ sudo nano local.rules
далі прописуємо правила реагування (рис. 7).

```
# This file intentionally does not come with signatures.  Put your local
# additions here.
alert icmp any any -> $HOME_NET any (msg: "ICMP Detected"; sid: 1000001; rev:1)
alert tcp any any -> any 22 (msg: "SSH Detected"; sid: 1000002; rev:1)
alert tcp any any -> any 443 (msg: "HTTPS Detected"; sid: 1000003; rev:1)
```

Рис. 7. Вигляд правил реагування на події

Перевіряємо конфігурацію командою:
~$ sudo snort -T -i ens18 -c /etc/snort/snort.conf
Якщо все дії здійснені правильно, повинні отримати звіт щодо роботи програмних компонентів Snort. На рис. 8 показано початок і кінець виводу результатів роботи програмного засобу Snort після конфігурування.

```
        --== Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
        --== Initialization Complete ==--

Snort successfully validated the configuration!
Snort exiting
```

Рис. 8. Звіт щодо конфігурації Snort

Після налаштування, виконуємо запуск IDS Snort для моніторингу трафіку.
Зазначена дія виконується наступною командою:
~$ sudo snort -A console -q -i ens18 -c /etc/snort/snort.conf -K ascii
За результатами запуску моніторингу відкривається рядок прослуховування мережевого інтерфейса Snort (рис. 9).

```
linaro@linaro-Standard-PC-i440FX-PIIX-1996:/etc/snort/rules$ sudo snort -A console -q -i ens18 -c /etc/snort/snort.conf -K ascii
[sudo] password for linaro:
```

Рис. 9. Рядок прослуховування мережевого інтерфейса Snort

Перевіримо як буде реагувати Snort на ICMP, SSH, HTTPS трафік, без застосування SSLH-проксі. Для перевірки протоколу ICMP, здійснимо перевірку зв'язку з віртуальною мащиною VM2 за допомоги команди ping. За результатами перевірки Snort виявляє «пінги» свого мережевого інтерфейсу (рис. 10).

```
linaro@linaro-Standard-PC-i440FX-PIIX-1996:/etc/snort/rules$ sudo snort -A console -q -i ens18 -c /etc/snort/snort.conf -K ascii
10/09-17:06:43.460845  [**] [1:1000001:1] ICMP Detected [**] [Priority: 0] {ICMP} 192.168.0.7 -> 192.168.0.18
10/09-17:06:43.460886  [**] [1:1000001:1] ICMP Detected [**] [Priority: 0] {ICMP} 192.168.0.18 -> 192.168.0.7
```

Рис. 10. Результати перевірки роботи протоколу ICMP засобами Snort

Використовуючи таку методику, перевіряємо відповідно підключення по протоколу SSH та протоколу HTTPS з VM1 або PC1 рис. 11, рис. 12.

```
linaro@linaro-Standard-PC-i440FX-PIIX-1996:/etc/snort/rules$ sudo snort -A console -q -i ens18 -c /etc/snort/snort.conf -K ascii
10/09-17:00:59.840341  [**] [1:1000002:1] SSH Detected [**] [Priority: 0] {TCP} 192.168.0.7:9467 -> 192.168.0.18:22
10/09-17:00:59.840808  [**] [1:1000002:1] SSH Detected [**] [Priority: 0] {TCP} 192.168.0.7:9467 -> 192.168.0.18:22
10/09-17:00:59.841912  [**] [1:1000002:1] SSH Detected [**] [Priority: 0] {TCP} 192.168.0.7:9467 -> 192.168.0.18:22
```

Рис. 11. Результати перевірки роботи протоколу SSH засобами Snort

```
linaro@linaro-Standard-PC-i440FX-PIIX-1996:/etc/snort/rules$ sudo snort -A console -q -i ens18 -c /etc/snort/snort.conf -K ascii
10/09-16:19:04.610315  [**] [1:1000003:1] HTTPS Detected [**] [Priority: 0] {TCP} 192.168.0.7:8005 -> 192.168.0.18:443
10/09-16:19:04.861257  [**] [1:1000003:1] HTTPS Detected [**] [Priority: 0] {TCP} 192.168.0.7:8006 -> 192.168.0.18:443
10/09-16:19:05.113471  [**] [1:1000003:1] HTTPS Detected [**] [Priority: 0] {TCP} 192.168.0.7:8005 -> 192.168.0.18:443
```

Рис. 12. Результати перевірки роботи протоколу HTTPS засобами Snort

Далі здійснюємо налаштування SSLH-проксі на порту 4444, і направлення вхідного трафіка на локальний інтерфейс 127.0.0.1, на якому Snort прослуховує порти 22, 443, 80 (рис. 13).

```
# binary to use: forked (sslh) or single-thread (sslh-select) version
# systemd users: don't forget to modify /lib/systemd/system/sslh.service
DAEMON=/usr/sbin/sslh

DAEMON_OPTS=" --user sslh --listen 0.0.0.0:4444 --ssh 127.0.0.1:22 --ssl 127.0.0.1:443 --http 127.0.0.1:80 --pidfile /var/run/sslh/sslh.pid"
```

Рис. 13. Перенаправлення і фільтрування трафіку в залежності від протоколу з використанням SSLH-проксі

Результати перевірки свідчать про те, що сервіс SSLH активний, це підтверджується роботою у фоновому режимі процесів 3008, 3009, 3031 рис. 14, рис. 15.

```
● sslh.service - SSL/SSH multiplexer
     Loaded: loaded (/lib/systemd/system/sslh.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2024-10-24 16:31:47 EEST; 10min ago
       Docs: man:sslh(8)
   Main PID: 3008 (sslh)
      Tasks: 3 (limit: 9439)
     Memory: 836.0K
        CPU: 106ms
     CGroup: /system.slice/sslh.service
             ├─3008 /usr/sbin/sslh --foreground --user sslh --listen 0.0.0.0 4444 --ssh 127.0.0.1 22 --tls 127.0.0.1 443 --http 127.0.0.1 80 --pidfile /var/run/sslh/sslh.pid
             ├─3009 /usr/sbin/sslh --foreground --user sslh --listen 0.0.0.0 4444 --ssh 127.0.0.1 22 --tls 127.0.0.1 443 --http 127.0.0.1 80 --pidfile /var/run/sslh/sslh.pid
             └─3031 /usr/sbin/sslh --foreground --user sslh --listen 0.0.0.0 4444 --ssh 127.0.0.1 22 --tls 127.0.0.1 443 --http 127.0.0.1 80 --pidfile /var/run/sslh/sslh.pid
```

Рис. 14. Вигляд активної роботи сервісу SSLH

Також можна перевірити наявність процесів, що запущені в системі, командою:
ps -af | grep sslh

```
linaro@linaro-Standard-PC-i440FX-PIIX-1996: $ ps -ef | grep sslh
sslh        3008       1  0 16:31 ?        00:00:00 /usr/sbin/sslh --foreground --user sslh --listen 0.0.0.0 4444 --ssh 127.0.0.1 22 --tls 127.0.0.1 443 --http 127.0.0.1 80 --pidfile /var/run/sslh/sslh.pid
sslh        3009    3008  0 16:31 ?        00:00:00 /usr/sbin/sslh --foreground --user sslh --listen 0.0.0.0 4444 --ssh 127.0.0.1 22 --tls 127.0.0.1 443 --http 127.0.0.1 80 --pidfile /var/run/sslh/sslh.pid
sslh        3031    3009  0 16:35 ?        00:00:00 /usr/sbin/sslh --foreground --user sslh --listen 0.0.0.0 4444 --ssh 127.0.0.1 22 --tls 127.0.0.1 443 --http 127.0.0.1 80 --pidfile /var/run/sslh/sslh.pid
```

Рис. 15. Результати перевірки наявності процесів в системі

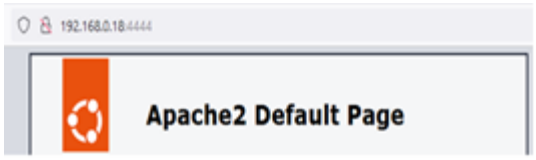

Рис. 16. Тестова сторінка веб-сервера Apache2 (HTTP/HTTPS трафік)

Тепер підключимось до HTTP/HTTPS веб-сервера на віртуальній машині VM2 з використанням порту 4444. Бачимо тестову сторінку веб-сервера Apache2 (рис. 16), що свідчить про роботу SSLH-проксі згідно наведеній на (рис. 1) топології взаємодії між SSLH-проксі і Snort, який відреагував на HTTP/HTTPS трафік (рис. 17).

```
linaro@linaro-Standard-PC-i440FX-PIIX-1996: $ sudo snort -A console -q -i ens18 -c /etc/snort/snort.conf -K ascii
10/24-16:57:51.103552  [**] [1:1000003:1] HTTP Detected [**] [Priority: 0] {TCP} 192.168.0.5:43607 -> 192.168.0.18:80
10/24-16:57:51.103930  [**] [1:1000003:1] HTTP Detected [**] [Priority: 0] {TCP} 192.168.0.5:43607 -> 192.168.0.18:80
10/24-16:58:22.725036  [**] [1:1000004:1] HTTPS Detected [**] [Priority: 0] {TCP} 192.168.0.5:43619 -> 192.168.0.18:443
10/24-16:58:22.981490  [**] [1:1000004:1] HTTPS Detected [**] [Priority: 0] {TCP} 192.168.0.5:43620 -> 192.168.0.18:443
```

Рис. 17. Результати роботи програмного забезпечення Snort під час аналізу трафіку з використанням протоколів HTTP/HTTPS

Аналогічно, при підключенні з VM1 або PC1 до VM2 за протоколом SSH через 4444 порт, процес роботи видно у виводі працюючого SSLH-проксі (рис. 18).

```
● sslh.service - SSL/SSH multiplexer
     Loaded: loaded (/lib/systemd/system/sslh.service; enabled; vendor preset: enabled)
     Active: active (running) since Thu 2024-10-24 16:31:47 EEST; 10min ago
       Docs: man:sslh(8)
   Main PID: 3008 (sslh)
      Tasks: 3 (limit: 9439)
     Memory: 836.0K
        CPU: 106ms
     CGroup: /system.slice/sslh.service
             ├─3008 /usr/sbin/sslh --foreground --user sslh --listen 0.0.0.0 4444 --ssh 127.0.0.1 22 --tls 127.0.0.1 443 --http 127.0.0.1 80 --pidfile /var/run/sslh/sslh.pid
             ├─3009 /usr/sbin/sslh --foreground --user sslh --listen 0.0.0.0 4444 --ssh 127.0.0.1 22 --tls 127.0.0.1 443 --http 127.0.0.1 80 --pidfile /var/run/sslh/sslh.pid
             └─3031 /usr/sbin/sslh --foreground --user sslh --listen 0.0.0.0 4444 --ssh 127.0.0.1 22 --tls 127.0.0.1 443 --http 127.0.0.1 80 --pidfile /var/run/sslh/sslh.pid

жов 24 16:34:08 linaro-Standard-PC-i440FX-PIIX-1996 sslh[3027]: tls:connection from 192.168.0.5:43144 to linaro-Standard-PC-i440FX-PIIX-1996:4444 forwarded from localhost:54494 to localhost:https
жов 24 16:35:18 linaro-Standard-PC-i440FX-PIIX-1996 sslh[3031]: ssh:connection from 192.168.0.5:43156 to linaro-Standard-PC-i440FX-PIIX-1996:4444 forwarded from localhost:35196 to localhost:ssh
жов 24 16:35:49 linaro-Standard-PC-i440FX-PIIX-1996 sslh[3118]: tls:connection from 192.168.0.5:43160 to linaro-Standard-PC-i440FX-PIIX-1996:4444 forwarded from localhost:59416 to localhost:https
жов 24 16:35:50 linaro-Standard-PC-i440FX-PIIX-1996 sslh[3119]: tls:connection from 192.168.0.5:43161 to linaro-Standard-PC-i440FX-PIIX-1996:4444 forwarded from localhost:59424 to localhost:https
жов 24 16:35:50 linaro-Standard-PC-i440FX-PIIX-1996 sslh[3120]: tls:connection from 192.168.0.5:43162 to linaro-Standard-PC-i440FX-PIIX-1996:4444 forwarded from localhost:59426 to localhost:https
жов 24 16:35:51 linaro-Standard-PC-i440FX-PIIX-1996 sslh[3121]: tls:connection from 192.168.0.5:43163 to linaro-Standard-PC-i440FX-PIIX-1996:4444 forwarded from localhost:59432 to localhost:https
жов 24 16:35:52 linaro-Standard-PC-i440FX-PIIX-1996 sslh[3122]: tls:connection from 192.168.0.5:43164 to linaro-Standard-PC-i440FX-PIIX-1996:4444 forwarded from localhost:59434 to localhost:https
жов 24 16:36:15 linaro-Standard-PC-i440FX-PIIX-1996 sslh[3123]: http connection from 192.168.0.5:43167 to linaro-Standard-PC-i440FX-PIIX-1995:4444 forwarded from localhost:52316 to localhost:http
жов 24 16:40:18 linaro-Standard-PC-i440FX-PIIX-1996 sslh[3131]: ssh:connection from 192.168.0.5:43192 to linaro-Standard-PC-i440FX-PIIX-1996:4444 forwarded from localhost:34198 to localhost:ssh
жов 24 16:41:32 linaro-Standard-PC-i440FX-PIIX-1996 sslh[3184]: ssh:connection from 192.168.0.5:43202 to linaro-Standard-PC-i440FX-PIIX-1996:4444 forwarded from localhost:35266 to localhost:ssh
```

Рис. 18. Процес мультиплексування у виводі працюючого SSLH-проксі

Висновки

У роботі був проведений аналіз роботи мультиплексора протоколів SSL/ SSH (SSLH-проксі) для організації доступу до віддаленої мережевої системи та механізм виконання моніторингу мережевого трафіка засобами IDS Snort. Дослідження було виконано засобами віртуалізації мережевих ресурсів. Для дослідження була використана мережева топологія рис. 1 з моделюванням доступу до віддалених ресурсів і аналізом мережевого трафіку засобами встановленої на них IDS Snort з використанням мультиплексора протоколів SSL/SSH (SSLH-проксі). За результатами дослідження можна зробити висновки, що отримані результати наочно демонструють механізм доступу до віддалених ресурсів по алгоритму мультиплексора SSL/SSH (SSLH-проксі), і алгоритми роботи правил IDS Snort.

Проведене дослідження дозволило продемонструвати переваги використання засобів віртуалізації і ефективного їх використання для багатокомпонентних мережевих топологій, значно спростити налаштування мережевої інфраструктури, а також дозволило здійснювати виконання лабораторної роботи в ізольованому середовищі.

Список літератури

1. Mike Rash, IDS signature matching with iptables, psad, and fwsnort https://www.usenix.org/system/files/login/articles/530-rash.pdf

2. Cryptography and Network Security: Principles and Practice, 7th Edition, ISBN 978-0-13-444428-4, by William Stallings published by Pearson Education © 2017.

3. Setting Up VirtualBox / Virtual Lab for Penetration Testing in Kali Linux / Backtrack | AmIRootYet. [Electronic resource]. – URL : https://www.amirootyet.com/post/setting-up-virtualbox-virtual-lab-for

4. SNORT Users Manual 2.9.16, [Electronic resource]. – URL : https://snort-org-site.s3.amazonaws.com/production/document_files/files/000/000/249/original/snort_manual.pdf

5. Snort 2.1 - Intrusion Detection, Second Edition. [Electronic resource]. – URL: https://www.academia.edu/4903498/Snort_2_1_Intrusion_Detection_Second_Edition

6. Sslh installation guide. [Electronic resource]. – URL: https://wiki.meurisse.org/wiki/sslh

7. Applicative protocol multiplexer. [Electronic resource]. – URL: https://packages.debian.org/ru/sid/sslh

8. Maitreyee Dutta, Ms. Shweta Sharma, Lab manual on snort - network intrusion detection system. [Electronic resource]. – URL: https://www.nitttrchd.ac.in/imee/Labmanuals/SNORT-%20Network%20Intrusion%20Detection%20System.pdf

9. Sebastien Simon, Snort Intrusion Detection System (IDS). [Electronic resource]. – URL: https://medium.com/@sebastienwebdev/snort-intrusion-detection-system-ids-2dd8f72123bb

O. Konovalov, Y. Dobryshyn, S. Sydorenko

MODELING ACCESS TO A REMOTE NETWORK SYSTEM WITH IDS SNORT USING AN SSL/SSH MULTIPLEXER

The article examines the use of an SSL/SSH multiplexer (SSLH proxy) for organizing access to a remote network system with Snort IDS in the preparation of cybersecurity courses. Utilizing the capabilities of the SSL/SSH protocol multiplexer (SSLH proxy), the principles and mechanisms for organizing access to a remote Linux Ubuntu operating system with the Snort IDS system deployed were explored. An analysis of the operation and configuration modes of Snort IDS was carried out to monitor such types of connections using different protocols. Examples of configuration files for both the SSL/SSH multiplexer (SSLH proxy) and the Snort IDS system were provided, and the operation of both software products was tested using different protocols and ports to organize remote access to the Linux Ubuntu 22.04 operating system. The testing was conducted using physical equipment with Windows 10 operating system, a Type 1 hypervisor, and virtual machines with Linux Ubuntu operating systems. The use of an SSL/SSH multiplexer (SSLH proxy) alongside Snort allows for the analysis of encrypted traffic, addressing key security challenges, such as detecting data leaks, malware, suspicious connections, and anomalous activity. This greatly enhances the ability to audit computer networks and equipment, improving the capability of intrusion detection systems to perform their functions in the face of modern threats, where much of the traffic is encrypted either for protection or to hide malicious activity.

The advantages of using virtualization to deploy a test network environment based on the SSL/SSH multiplexer and the Snort intrusion detection system are also highlighted. Snort is not

only an intrusion detector, but also a packet logger and sniffer. However, its most important feature is intrusion detection. Snort is rule-based, and you can download basic rules from the Snort website and configure them according to your specific needs. Snort performs intrusion detection using methods based on Anomaly and Signature. It is also useful for deep analysis of the data it collects. Moreover, the basic Snort rules can be used to detect a wide variety of events, including CGI attacks, buffer overflow attacks, and Stealth port scans. For studying the operation of the SSL/SSH protocol multiplexer and Snort IDS as part of organizing a laboratory exercise, a lab assignment was chosen to simulate the deployment of the SSL/SSH protocol multiplexer and analyze network traffic using Snort IDS tools.

Keywords: SSL/SSH protocol multiplexer, Snort IDS, remote access, network traffic analysis.