МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Кваліфікаційна наукова
праця на правах рукопису

АНДЕРС КАРЛССОН

УДК 621.391

ДИСЕРТАЦІЯ

МОДЕЛЬ ТА МЕТОД ВИЯВЛЕННЯ НИЗЬКОІНТЕНСИВНИХ МЕРЕЖЕВИХ АТАК НА ПРИКЛАДНОМУ РІВНІ

Спеціальність: 05.12.02 – телекомунікаційні системи та мережі
05 «Технічні науки»
Подається на здобуття наукового ступеня кандидата технічних наук

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело
_____А. Карлссон

Науковий керівник: Дуравкін Євген Володимирович, доктор технічних наук, доцент

Ідентичність всіх примірників дисертації засвідчую:
Вчений секретар
спеціалізованої вченої ради /О.Б. Ткачова/

Харків – 2017

АНОТАЦІЯ

Андерс Карлссон. Модель та метод виявлення низькоінтенсивних атак на прикладному рівні – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня кандидата технічних наук (доктора філософії) за спеціальністю 05.12.02 – телекомунікаційні системи та мережі. – Харківський національний університет радіоелектроніки, – Міністерство освіти і науки України, – Харків, – 2017.

Дисертаційна робота присвячена вирішенню актуальної науково-прикладної задачі, яка пов'язана з забезпеченням гарантованого рівня якості обслуговування у мільтисервісних мережах за рахунок розробки та впровадження нового методу виявлення низькоінтенсивних атак типу «відмова в обслуговуванні» на прикладному рівні.

У результаті проведеного в роботі аналізу встановлено, що зростаюча популярність технології хмарних обчислень, як складової інфокомунікаційних систем привертає увагу не тільки кінцевих користувачів та розробників клієнтських послуг, але і зловмисників. Відсутність універсальних стандартів інформаційної безпеки в хмарних системах є перешкодою для розвитку і освоєння цієї області. Однією з ключових проблем сучасних інфокомунікаційних систем є захист послуг, що надаються від атак типу відмова в обслуговуванні (DOS-атак). Основною особливістю даного типу атак є порівняно низька інтенсивність трафіку з атаками на мережевому рівні. Аналіз профілю інтенсивності такого трафіку не містить аномалій. Відрізнити трафік, що генерується під час таких атак від законного трафіку, досить складно.

Встановлено, загально відомі засоби та методи виявлення атак на відмову в обслуговуванні базуються на детектуванні аномалій трафіку, характерних для атак. В той же час атаки на відмову в обслуговуванні прикладного рівня не вимагають генерації великого обсягу трафіку тому атаки цього типу досить важно ідентифікуються звичайними системами. Таким чином, сучасні рішення захисту інформації не дозволяють ефективно виявляти DoS-атаки прикладного

рівня, що викликає значне зростання їх кількості. На відміну від атак третього і четвертого рівнів атаки прикладного рівня не вимагають великої атакуючої бот-мережі і надійно «відкидають» ресурс, що атакується, залишаючись практично невидимими для спеціалізованого обладнання, встановленого провайдером.

З метою підвищення доступності послуг, що надаються мультисервісною мережею за рахунок вдосконалення існуючих та розробки нових методів виявлення мережевих атак прикладного рівня у дисертаційній роботі поставлено та розв'язано наступні наукові задачі:

отримано розподіл ймовірностей стану web-серверу під час реалізації Slow-http атак різного типу. Дослідження природи Slow-http атак показали, що в процесі реалізації потік заявок з джерела атаки можна вважати найпростішим. Про це свідчить те, що досліджуваний потік заявок володіє трьома властивостями, характерними найпростішого потоку подій: стаціонарність, ординарність і відсутність наслідків;

розроблено модель виявлення низькоінтенсивних мережевих атак, на відмову в обслуговуванні прикладного рівня. Розроблений метод дозволяє виявити факт виконання атаки, визначити джерело нападу і блокувати зловмисний трафік. Метод базується на аналізі поведінки сервера в нормальному режимі і при реалізації різновидів slow-http атак;

розроблено модель аналізу навантаження серверів додатків на основі графів вірогідності та часу. Модель базується на використанні ймовірносно-часових функцій та базується на інформації, що попередньо отримана за допомогою моделі станів web-серверу. Даний математичний апарат дозволяє зв'язати модель станів системи, що отримана у попередньому розділі з динамікою функціонування системи;

розроблено метод виявлення та класифікації атак на відмову в обслуговуванні прикладного рівня на web-сервер. Розроблений метод дозволяє виявити факт виконання атаки, визначити джерело нападу та блокувати зловмисний трафік.

Проведено експеримент з реалізації slow-http атаки, який дозволить виявити характерні особливості кожного типу атаки і висунути припущення

щодо можливості їх виявлення. В основі запропонованої у експерименті системи захисту лежить аналіз поведінки сервера в нормальному режимі та при реалізації різновидів типу Slow-http атак. Процес виявлення атаки реалізовано на основі моделі Маркова за поведінкою сервера, параметрами моделі є статистичні характеристики вхідного, вихідного трафіку, а також динаміки серверу ресурсного використання. Перевага запропонованої системи полягає в тому, що вона дозволяє виявляти атаку на відмову в обслуговуванні сервера, що дає можливість своєчасно активувати відповідні механізми захисту.

Результати аналізу експерименту дозволили ідентифікувати найбільш чутливі параметри для атак повільного нападу, а також встановити їх порогові значення в залежності від таких показників, як потужності каналу, продуктивності обладнання та параметрів конфігурації сервера.

Розроблені математичні моделі, що дозволяють розраховувати розподіл ймовірностей станів web-серверу в залежності від параметрів вхідного та вихідного потоків можуть буди використані як ядро системи виявлення та протидії мережевим атакам на web-сервери.

Список публікацій здобувача:

1. Anders Carlsson ИНФРАСТРУКТУРА PenTestING И УПРАВЛЕНИЯ УЯЗВИМОСТЬЮ/ Хаханов Владимир Иванович, Чумаченко Светлана Викторовна, Anders Carlsson // АСУ и приборы автоматики. – 2012. – №160 – С. 36-51

2. Anders Carlsson Модели управления уязвимостью /Хаханов Владимир Иванович, Anders Carlsson, Чумаченко Светлана Викторовна, Бутенко Сергей Александрович // АСУ и приборы автоматики. – 2012. – №161. – С.10-24

3. Carlsson A. Analysis of realization and method of detecting low-intensity HTTP-attacks [Электроний ресурс] / A. Carlsson, E.V. Duravkin, A.S. Loktionova // Проблеми телекомунікацій. – 2013. – № 3 (12). – С. 61-70. – Режим доступу до журналу: http://pt.journal.kh.ua/2013/3/1/133_carlsson_ attack.pdf

4. Carlsson A. A. Analysis of realization and method of detecting low-intensity HTTP-attacks. Part 2. Method of detecting Slow HTTP attacks [Электроний

ресурс] / A.A. Carlsson, I.V. Duravkin, A.S. Loktionova // Проблеми телекомунікацій. – 2014. – № 1 (13). – С. 96-100. – Режим доступу до журналу: http://pt.journal.kh.ua/2014/1/1/ 141_carlsson_attack.pdf.

5. Carlsson A. A. Method of slow-attack detection / Carlsson Anders, I. V. Duravkin, A. S. Loktionova // Системи обробки інформації. — 2014. — № 8. — С. 102-106.

6. Carlsson Anders. Detecting cyber threats through social network analysis/ Carlsson Anders, Kirichenko Lyudmyla, Radivilova Tamara // SocioEconomic Challenges, – №1(1), – 2017. –p. 20-34.

7. Anders Carlsson. Model of network attack on the cloud platform OpenStack/ Anders Carlsson // In proc. of Second International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T'2015), Kharkiv, Ukraine. – 2015. – p. 30-33

8. Anders Carlsson Infrastructure of pentesting and vulnerability management // Vladimir Hahanov, Anders Carlsson, Svetlana Chumachenko/ In Proc. of conference, Kharkov, Ukraine,ISBN 0135-17, 10 Sep. 2012, – P. 10-24.

9. Wajeb Gharibi, Hahanov V. I., Anders Carlsson, Hahanova I. V., Filippenko I.V. Quantum technology for analysis and testing computing systems // In Proc. of IEEE EastWest Design & Test Symposium (EWDTS'2013), Rostov-on-Don, Russia, Sep. 27-30 , 2013.

10. Anders Carlsson; Rune Gustavsson: "Resilient Smart Grids."// In Proc. of First International Scientific-Practical Conference Problems of Infocommunications Science and Technology, Kharkov, Ukraine, Oct. 14-17, 2014

11. Alexander Adamov, Vladimir Hahanov, Anders Carlsson. Discovering New Indicators for Botnet Traffic Detection / Alexander Adamov, Vladimir Hahanov, Anders Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2014), September 26–29, 2014, Kiev, Ukraine. – Kiev, 2014. – P. 281–285.

12. Anders Carlsson, Rune Gustavsson. Resilient Smart Grids //In Proc. of First International Scientific-Practical Conference Problems of Infocommunications Science and Technology, Kharkov, Ukraine, Oct. 14-17, 2014 PIC_S&T– pp. 79-82

13. Adamov A, Carlsson A. A Sandboxing Method to Protect Cloud Cyberspace / Adamov A, Carlsson A // Proceedings of IEEE East-West Design & Test Test Symposium (EWDTS'2015), September 27-30, 2015, Batumi, Georgia – P. 180–183.

14. Anders Carlsson. Model of network attack on the cloud platform OpenStack / Anders Carlsson // In proc. of Second International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T'2015), Kharkiv, Ukraine, Oct. 13 -15, – 2015

15. Kurt Tutschku, Vida Ahmadi Mehri, Anders Carlsson, Krishna Varaynya Chiukula, Johan Christenson. On Resource Description Capabilities of On-Board Tools for Resource Management in Cloud Networking and NFV Infrastructures / Kurt Tutschku, Vida Ahmadi Mehri, Anders Carlsson, Krishna Varaynya Chiukula, Johan Christenson // In Proc. of First IEEE International Workshop on Orchestration for Software Defined Infrastructures (co-located with IEEE ICC 2016), Kuala Lumpur, Malaysia, May 23 - 27, 2016.

16. Kurt Tutschku, Vida Ahmadi Mehri, Anders Carlsson. Towards Multi-layer Resource Management in Cloud Networking and NFV Infrastructures / Kurt Tutschku, Vida Ahmadi Mehri, Anders Carlsson // In Proc. of 12th Swedish National Computer Networking Workshop (SNCNW), Sundsvall, Sweden, Jun. 1-2, 2016.

17. Alexander Adamov, Anders Carlsson. Cloud incident response model / Alexander Adamov, Anders Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2016), October 14–17, 2016, Yerevan, Armenia. – P. 250–253.

18. Ievgeniia Kuzminykh, Arkadii Snihurov, Anders Carlsson. Testing of communication range in ZigBee technology / Ievgeniia Kuzminykh, Arkadii Snihurov, Anders Carlsson // In Proc. of 14th International Conference on The Experience of Designing and Application Systems in Microelectronics (CADSM'2017), Polyana, Svalyava (Zakarpattya), Ukraine, Feb. 21 – 25, 2017.

19. Alexander Adamov, Anders Carlsson. The State of Ransomware. Trends and Mitigation Techniques / Alexander Adamov, Anders Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2017), Sep 29 – Oct 2, 2017, Belgrad, Serbia. – P. 121–128.

ЗМІСТ

2.1.2. Детерміновані підходи на основі правил

2.1.3. Підхід на основі експертної оцінки

2.1.4. Підхід на основі правил нечіткої логіки

2.1.5. Підхід на основі нейронних мереж

2.1.6. Підхід заснований на генної інженерії

2.1.7. Підхід на основі Марковських ланцюгів

2.2. Формальна модель поведінки сервісу через набір станів

2.3. Модель виявлення Slow HTTP атак

2.4. Аналіз адекватності моделі виявлення Slow HTTP атак

2.5. Оцінка ефективності моделі

2.6. Висновки по розділу

3. Розробка моделі прогнозування часу переходу сервісу у стан «відмова в обслуговуванні»

3.1. Аналіз поведінки сервісу і модель атаки

3.1.1. Аналіз поведінки сервісу

3.1.2. Метод збору інформації про реалізацію атаки

3.2. Метод ймовірносно-часових графів

3.2.1. Метод еквівалентних перетворень ймовірносно-часових графів

3.2.2. Метод аналізу ймовірносно-часових характеристик на «неприводимих» ВВГ.

3.3. Розробка методу прогнозування часу реалізації slow-http атаки

3.3.1. Метод оцінки тимчасових характеристик при реалізації slow-http атаки

3.3.2. Модель аналізу динаміки slow-http атаки

3.4. Висновки по розділу

4. Методи виявлення та класифікації Slow HTTP атак

4.1. Модель безпеки процесу обробки даних в хмарних системах.

4.2. Виявляти й визначати Slow-HTTP атак на хмарну структуру OpenStack

4.2.1. Аналіз архітектури OpenStack

4.2.2. Розробка методу виявлення і класифікації Slow-HTTP атак

4.3. Система виявлення та запобігання Slow-HTTP атак

4.3.1. Класифікація атакуючого трафіку.

4.4. Аналіз ефективності методу виявлення і класифікації Slow-HTTP атак.

4.5. Висновки по розділу

Висновки

Список використаних джерел

Перелік умовних скорочень

ACSR - Algebra Of Communicating Shared Resources

API - Application Programming Interface

ETSI - European Telecommunications Standards Institute

IEEE - Institute of Electrical and Electronics Engineers

IETF - Internet Engineering Task Force

IP - Internet Protocol

IRTF - Internet Research Task Force

ITU-T - International Telecommunication Union-Telecommunication

LLDP - Logical Link Discovery Protocol

MIB - Management Information Base

NDB - Non-Directional Beacon

OFDP - OpenFlow Discovery Protocol

ONF - Open Network Foundation

QoS - Qulaty of Service

RTT - Round-Trip Time

SDL - Specification and Description Language

SDN - Software-Defined Networking

SDNRG - Software-Defined Networking Research Group

TCP - Transmission Control Protocol

TLV - Tag Length Value

ToS - Type of Service

TTL - Time To Live

UDP - User Datagram Protocol

UML - Unified Modeling Language

ВСТУП

Концепція безпеки мережевої інфраструктури є однією з основних вимог, які повинні бути реалізовані в усіх сучасних інфокомунікаційних системах. Важливість цієї концепції зростає зі збільшенням проникнення інфокомунікаційних систем в різні аспекти життя суспільства. Широке і зростаюче проникнення технології хмарних обчислень, як складової інфокомунікаційних систем привертає увагу не тільки кінцевих користувачів та розробників клієнтських послуг, але і зловмисників. Відсутність універсальних стандартів інформаційної безпеки в хмарних системах є перешкодою для розвитку і освоєння цієї області. Однією з ключових проблем сучасних інфокомунікаційних систем є захист послуг, що надаються від атак типу відмова в обслуговуванні (DOS-атак).

Аналіз статистики компаній, що забезпечують захист інформаційних ресурсів мережі, показав значне збільшення обсягу і складності DOS-атак за останні кілька років.

Зусилля провайдерів в усьому світі спочатку були зосереджені на очищенні каналів від трафіку, що лише забирає пропускну здатність каналу зв’язку. Наприклад, багато інтернет-провайдерів використовували спеціалізовані рішення: встановлення та налаштування брандмауера, маршрутизація в «чорні діри», використання систем виявлення вторгнень (IDS), списки контролю доступу та інш. Спеціалізовані засоби захисту від DOS-атак мережевого рівня основані на аналізі трафіку і виявленні аномалій в його структурі. Відповідно до цього підходу будуються профілі трафіку в різних умовах роботи і виконується пошук аномалій в спостережуваному діапазоні. В цьому випадку аномалія трафіку є відхилення характеристик від статистичних значень. Цей підхід заснований на використанні статистичних методів, вівлет аналізі, методах підпису, кластерному аналізі та інш. Вказані методи дозволяють ефективно розпізнавати та обробляти лавинні DoS-атаки мережевих і транспортних рівнів, спрямовані на заповнення пропускної здатності каналу

(Smurf, UDP-повінь і т. д.) і перевищення нормального навантаження окремих вузлів (SYN-flood, Teardrop, Ping смерті і т. д.).

У той же час дані рішення не дозволяють ефективно виявляти DoS-атаки прикладного рівня, що викликає значне зростання їх кількості. На відміну від атак третього і четвертого рівнів атаки прикладного рівня не вимагають великої атакуючої бот-мережі і надійно «відкидають» атакуємий ресурс, залишаючись практично невидимими для спеціалізованого обладнання, встановленого провайдером. Основною особливістю даного типу атак є порівняно низька інтенсивність трафіку з атаками на мережевому рівні. Аналіз профілю інтенсивності такого трафіку не містить аномалій. Відрізнити трафік, що генерується під час таких атак від законного трафіку, досить складно.

Основні зусилля по впровадженню таких атак – це відкриття з'єднання з сервером без відправки навіть одного байту. Відкриття з'єднання і очікування відповіді не вимагає майже ніяких ресурсів від зловмисника, але він завжди пов'язує один серверний процес, який чекає виконання запиту. Сервер буде чекати закінчення витоку часу очікування, а потім буде закрито.

Відкриття тільки одного з'єднання не призведе до значних пошкоджень, але одночасне відкриття сотень підключень займе всі доступні серверні процеси. При досягненні максимальної кількості процесів (а їх набагато менш ніж пропускна здатність мережі на мережевому рівні) сервер реєструє цю подію в журналі помилок («сервер досяг максимальної кількості запитів (Max Clients), розгляне можливість збільшення кількості Max Clients») і почне зберігати нові підключення в черзі. Якщо відкриття нових з'єднань триватиме з високою швидкістю, звичайні запити не будуть обслуговані. Якщо відкриття цих з'єднань триватиме з ще більшою швидкістю, черга буде переповнюватися, що призведе до відмови від нових з'єднань.

Завданнями розробки та впровадження засобів захисту телекомунікаційних мереж від DOS-атак займаються крупні IT-компанії, такі як Cisco, HP, IBM, Juniper та академічні інститути - ONF, IRTF, IETF, ETSI, SDNRG. Методам дослідження та розробки системи управляння, зокрема, моделям та

методам аналізу коректності поведінки та розподілу мережевих ресурсів присвячено ряд робіт дослідників у всьому світі.

Виходячи з цього, науково-прикладна задача, що полягає у підвищенні доступності послуг, що надаються мультисервісною мережею за рахунок вдосконалення існуючих та розробки нових методів виявлення мережевих атак, а отже й тема дисертаційної роботи «Модель та метод виявлення низькоінтенсивних атак прикладного рівня», що спрямована на вирішення зазначеної задачі, є актуальною.

Метою дисертаційної роботи є підвищення доступності послуг, що надаються мультисервісною мережею за рахунок вдосконалення існуючих та розробки нових методів виявлення мережевих атак прикладного рівня.

Для досягнення поставленої мети у дисертаційній роботі розв’язано наступні наукові задачі:

1. Аналіз особливостей реалізації низькоінтенсивних атак на відмову в обслуговуванні;

2. Відбір набору конкретних характеристик для кожного типу низькоінтенсивних атак на відмову в обслуговуванні;

3. Розробка математичної моделі web-сервера при реалізації низькоінтенсивних атак на відмову в обслуговуванні;

4. Розробка математичної моделі для прогнозування стану сервісу, що надається на рівні додатків, з урахуванням одночасного обслуговування декількох запитів;

5. Розробка методу виявлення та класифікації низькоінтенсивних атак на відмову в обслуговуванні на web-сервіси (Slow-http атак).

Об’єкт дослідження – процес виявлення мережевих атак на відмову в обслуговуванні на прикладному рівні.

Предмет дослідження – моделі та методи виявлення низькоінтенсивних атак на відмову в обслуговуванні на web-сервіси.

Методи дослідження. Підчас розв’язання поставлених задач, зокрема, при оцінці взаємодії мережевих елементів у процесі надання комплексних сервісів було використано положення теорії управління багаторівневими

системами; положення теорії множин використано під час аналізу та розробки моделі web-серверу; теорія масового обслуговування та теорія графів – під час вирішення задачи з розробки моделі низькоінтенсивної мережевої атаки, та методи аналізу розподілу ресурсів - під час оцінки коректності та ефективності розподілу мережевих ресурсів; методи імітаційного моделювання та математичної статистики - під час проведення та оцінки результатів експериментального дослідження.

Наукова новизна отриманих результатів. В ході розв'язання наукової задачі було отримано наступні нові наукові результати:

1. Вперше розроблено модель виявлення низькоінтенсивних мережевих атак, на відмову в обслуговуванні прикладного рівня. Використання ланцюгів Маркова для подібного виявлення мережевих атак є новизною запропонованого підходу. Розроблена модель дає змогу аналізувати поведінку сервера, що атакується та обчислити ймовірність переходу web-серверу у стан «відмова в обслуговуванні». Застосування розробленої моделі дає можливість запровадити попереджувальні кроки та запобігти стані «відмови в обслуговуванні».

2. Вперше розроблено модель аналізу навантаження серверів додатків на основі графів вірогідності та часу. Застосування розробленої моделі дає можливість обчислювати час переходу сервера до стану «відмови в обслуговуванні» та оцінити динаміку впливу мережевих атак і вибрати контрзаходи для зниження його ефективності.

3. Вперше, розроблено метод виявлення та класифікації низькоінтенсивних атак типу «відмова в обслуговуванні» на web-сервіси. Розроблений метод дає можливість генерувати механізми зниження ефективності впливу подібних атак і, як наслідок, збільшити доступність послуг в мультисервісних мережах.

Обґрунтованість і достовірність отриманих в роботі нових наукових результатів забезпечена та підтверджена коректним використанням ключових положень добре відомого та апробованого математичного апарату – ланцюгів Маркова, теорії управління багаторівневими системами, теорії множин, теорії

графів та поширеними підходами до процесу реплікації та методів балансування навантаження.

Наукове значення. Запропоновані у дисертаційному дослідженні методи та моделі дозволяють підвищити якість надання сервісів у мультисервісній мережі, а саме підвищити доступність та захищеність від мережевих атак типу «відмова в обслуговуванні» що реалізуються на прикладному рівні.

Результати дисертаційної роботи також можуть бути рекомендовані до використання при проектуванні та вдосконаленні конвергентних телекомунікаційних систем, зокрема мультисервісних хмарних систем. Запропоновані методи та моделі можуть бути використані як науково-методична база для подальших досліджень функціонування та надання сервісів у різних типах мультисервісних систем, зокрема тих, що реалізуються у хмарному середовищі.

Практична значимість результатів досліджень полягає в тому, що запропоновані математичні моделі і методи можуть бути використані підчас розробки, підтримки, проектування та впровадження різноманітних мультисервісних хмарних систем. Запропонований метод був використаний для захисту хмарної лабораторії ReSeLa, та застосовується для підготовки фахівців в області інформаційної безпеки. Матеріали дисертаційної роботи використано в навчальному процесі кафедри інфокомунікаційних систем ХНУРЕ в курсі «Методи колективного захисту інформації».

Особистий внесок здобувача. Усі основні наукові результати, що висвітлено в дисертаційній роботі, здобувач отримав самостійно. Крім того, в роботі [1] автором особисто розроблено імітаційну модель низькоінтенсивної атаки прикладного рівня на хмарне середовище. В роботі [2] автором особисто розроблена модель поведінки web-серверу на базі ланцюгів Маркова. В роботі [3] автором розроблена модель, що дозволяє розрахувати час переходу web-сервера у стан «відмова в обслуговуванні». В роботі [4] автором розроблено метод виявлення та класифікації Slow-http атак на хмарні системи. В роботі [5] автором розроблено модель Slow-http атаки на соціальні мережі.

Апробація основних положень дисертаційної роботи була проведена у ході п’яти конференцій та п’яти форумів, а саме EastWest Design & Test Symposium (EWDTS) 2013, First International Scientific-Practical Conference Problems of Infocommunications Science and Technology 2014, IEEE East-West Design & Test Symposium (EWDTS’2014), First International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC_S&T 2014), East-West Design &Test Test Symposium (EWDTS’2015), First IEEE International Workshop on Orchestration for Software Defined Infrastructures (co-located with IEEE ICC 2016), East-West Design & Test Symposium (EWDTS’2016), East-West Design & Test Symposium (EWDTS’2017)

Публікації. Основні результати дисертаційної роботи опубліковані в п’ятнадцяти наукових працях: одна стаття у закордонному фаховому журналі [14], п’ять статей у фахових науково-технічних журналах та збірках наукових праць [1-4, 14]. Апробація результатів дисертації проходила в ході десяти доповідей на міжнародних науково-технічних конференціях [9-13, 15-17] які проходили під егідою IEEE та індексуються в міжнародних наукометричних базах Scopus та IEEE Xplore Digital Library.

Структура та обсяг дисертації. Дисертація складається зі вступу та чотирьох розділів. Загальний обсяг роботи становить 130 сторінок, в тому числі 110 сторінок основного тексту, 40 рисунків та 8 таблиць на 13 сторінках. Список використаних джерел містить 110 найменувань, викладених на 10 сторінках.

РАЗДЕЛ 1 АНАЛИЗ ОСОБЕННОСТЕЙ РЕАЛИЗАЦИИ АТАК ТИПА «ОТКАЗ В ОБСЛУЖИВАНИИ»

1.1 Анализ атак на отказ в обслуживании в облачной среде

Один из основных принципов, реализуемых в процессе использования современных информационных технологий, – принцип системной безопасности.

Актуальность данного принципа возрастает по мере развития информационных технологий, в рамках которых происходит выделение процессов порождения информации, ее отображения, хранения и обработки в отдельное производство, независимое от остальных процессов. Речь идет в первую очередь о распределенных базах данных, распределенной обработке информации, GRID-технологиях, SOA-архитектурах и, конечно, облачных вычислениях как модели предоставления по требованию удобного сетевого доступа к разделяемому пулу конфигурируемых ресурсов обработки данных (сети, серверы, ресурсы хранения, приложения и сервисы). Как правило, говорят о трех облачных моделях: инфраструктура как услуга (IaaS), платформа как услуга (PaaS), приложение как услуга (SaaS). В зависимости от варианта реализации с точки зрения категорий потребителей услуг облако может быть публичным, корпоративным или частным.

Широкое распространение облачных технологий привлекло к ним внимание не только конечных пользователей сервисов, но и злоумышленников. Отсутствие общепризнанных стандартов информационной безопасности в облачных системах является сдерживающим фактором развития в данной области.

В тоже время нельзя не заметить значительные усилия прилагаемые организацией OWASP и др. Однако они в большей степени относятся к web-составляющей облака и часто носят локальный не систематизированный характер.

Атаки на облачные системы можно условно разделить на два типа:

- традиционные атаки связанные с уязвимостями сетевых протоколов безопасности и программного обеспечения;

- функциональные атаки на компоненты облака. Данный тип атак основывается на особенностях используемой архитектуры облачного решения.

Атаки первого типа являются достаточно хорошо изученными и в настоящее время уже существует достаточно большое количество частных решений, направленных на борьбу с ними. Это и потоковые антивирусы, и средства шифрования, организации VPN и т.п.

В тоже время противодействие атакам второго типа полностью лежит на провайдерах облака и зачастую носит недостаточный характер. Свидетельством чему являются отказы сервисов и потеря данных.

Одним из вариантов атак второго типа является DOS-атака на компоненты облака.

Анализ архитектуры OpenStack показывает, что основным протоколом взаимодействия между компонентами системы является HTTP и технология REST, использующая его в качестве транспорта (рис. 1.1).

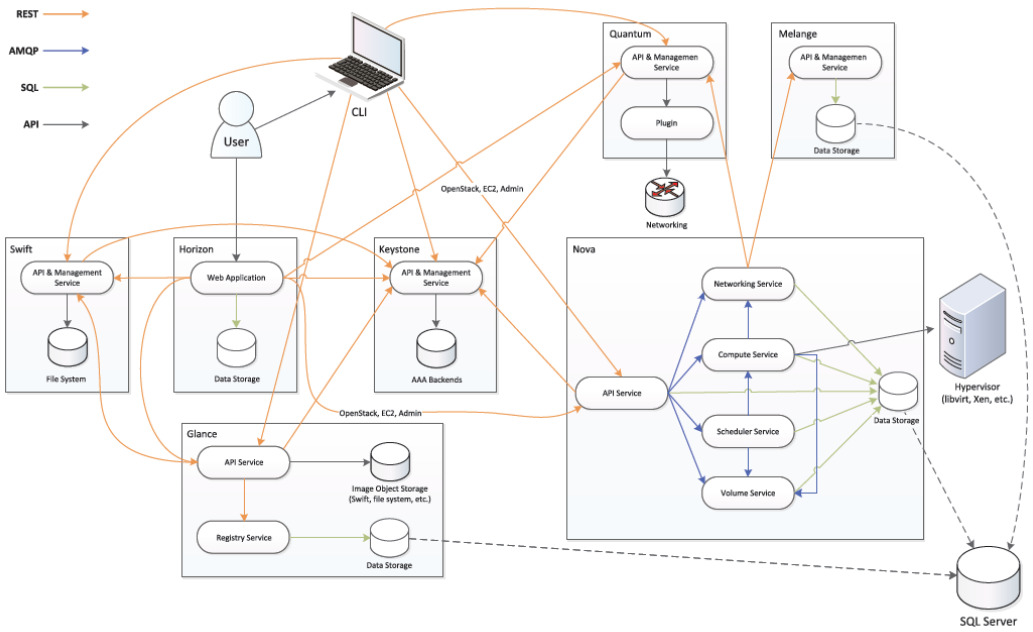


Рис. 1.1. Архитектура облачной инфраструктуры на базе OpenStack

1.2 Классификация известных атак типа «отказ в обслуживании»

Как правило, нарушение функционирования сети или системы осуществляется намного проще, чем получение доступа к ним. Сетевые протоколы типа TCP/IP были разработаны для применения в открытом и доверенном сообществе пользователей, и текущая версия 4 унаследовала все уязвимости своих предшественников. Кроме того, многие операционные системы и сетевые устройства имеют различные уязвимости в используемой реализации сетевого стека, что значительно снижает их способность противостоять DoS-атакам. Поскольку существует множество средств, для реализации DoS-атак, важным аспектом является формализация их по набору общих признаков, для упрощения идентификации атак, а так же последующего их выявления и предотвращения.

Единой и общепринятой классификации DoS - атак на сегодняшний день не существует. Условно специалисты выделяют несколько методик, наиболее распространенных среди злоумышленников:

1.2.1 Атаки, способствующие насыщению полосы пропускания

Целью атак данного типа является заполнение атакуемой сети большими объемами трафика. Необходимым условием для их осуществления является наличие у злоумышленника канала, с большим объемом, чем у атакуемого хоста. Атакующий отсылает на сервер множество запросов, насколько позволяет плотность его канала, забивая полосу пропускания атакуемой машины. Забитая линия не может пропустить к серверу еще какие-либо другие запросы, и информация с сервера становится временно недоступной для пользователей.

Самым распространенным представителем атак данного типа является флуд (англ. flood — «наводнение», «переполнение») — атака, целью которой является отказ в работе системы из-за исчерпания системных ресурсов процессора, памяти или каналов связи. Как правило, сценарий реализации флуда основан на заполнении каналов большим объемом трафика или сформированных в неправильном формате запросов к компьютерной системе или сетевому оборудованию [2]. Выделяется несколько разновидностей рассматриваемых атак:

1) Ping -flood.

Суть атаки заключается в заполнении полосы пропускания при помощи ping-запросов. Злоумышленник посылает «ICMP echo request»-пакеты с исходным адресом атакуемого хоста, на широковещательные адреса крупных сетей. В результате каждая из машин ответит на этот запрос, и хост-отправитель получит больше количество ответов. Рассылка множества «broadcast-echo» - запросов от имени атакуемого хоста на широковещательные адреса крупных сетей, может вызвать резкое переполнение канала атакуемого.

Ключевой особенностью атак данного типа является возможность их осуществления с помощью программ и утилит, входящих в состав домашних/офисных версий операционных систем, что значительно упрощает условия их реализации.

Предпосылкой для обнаружения Ping-flood атак является резкое изменение параметров сервера, таких как возрастание нагрузки на сеть (или канал), и превышение нормального значения количества специфических пакетов (таких, как ICMP)[3].

2) HTTP- флуд.

Суть данного типа атаки заключается в том, что атакующий шлет маленький по объему HTTP-пакет с заданными параметрами, требующими от сервера ответ большого размера. В качестве защиты от ответного HTTP-трафика, злоумышленник производит подмену своего ip-адреса на ip-адрес некого узла в сети[4].

3) Smurf-атака (ICMP - флуд).

Суть атаки типа Smurf заключается в передаче в сеть широковещательных ICMP запросов от имени атакуемого. В результате компьютеры, принявшие такие широковещательные пакеты, отвечают компьютеру-жертве, что приводит к существенному снижение пропускной способности канала связи и, в ряде случаев, к полной изоляции атакуемой сети[3].

4) Атака Fraggle (UDP-флуд).

Атака Fraggle (англ. fraggle attack – «осколочная граната».) является полным аналогом атаки Smurf, только вместо ICMP пакетов используются

пакеты UDP, отсюда второе название данного типа атак UDP-flood. Принцип действия этой атаки заключается в том, что на седьмой порт атакуемого отправляются echo-команды по широковещательному запросу. Далее подменяется IP-адрес злоумышленника на ip-адрес атакуемого, который получает множество ответных сообщений, количество которых зависит от числа узлов в сети. Данная атака приводит к насыщению полосы пропускания и полному отказу в обслуживании сервера[5].

5) SYN-флуд.

Суть атаки заключается в отправке большого количества SYN-запросов за достаточно короткий срок, переполняя очередь подключений на web-сервере. Далее атакующий игнорирует отправку пакетов SYN+ACK. либо подделывает заголовок пакета таким образом, что ответный SYN+ACK отправляется на несуществующий адрес. В очереди подключений появляются полуоткрытые соединения, ожидающие подтверждения от клиента. По истечении определенного тайм-аута эти подключения отбрасываются. Задача злоумышленника заключается в том, чтобы поддерживать очередь заполненной, не допуская новых подключений. В следствии, легитимные клиенты не могут установить связь, либо устанавливают её с существенными задержками. Атака основывается на уязвимости ограничения ресурсов операционной системы для полуоткрытых соединений, описанной в 1996 году группой CERT[1], согласно которой очередь для таких подключений была очень короткой, а тайм-аут подключений достаточно продолжительным (по RFC 1122 — 3 минуты).

Для противодействия атакам типа «насыщение полосы пропускания» возможно применение следующих мер:

– блокирование трафика с отдельных узлов и сетей;

– отключение ответов на ICMP-запросы (отключение соответствующих служб или предотвращение отклика на определенный тип сообщения) на web-сервере;

– снижение приоритета обработки ICMP-сообщений (при этом весь остальной трафик обрабатывается в обычном порядке, а ICMP-запросы

обрабатываются по остаточному принципу; в случае перегрузки ICMP-сообщениями часть из них игнорируется);

– отбрасывание или фильтрация ICMP-трафика средствами межсетевого экрана;

– увеличение очереди обрабатываемых подключений;

– использование SYN cookie;

– ограничение запросов на новые подключения от конкретного источника за определенный промежуток времени[6].

1.2.2. Атаки, возникающие в результате ошибок программирования

Реализация данного типа атак возможна, используя уязвимые места, ошибки и недокументированные функции операционных систем, программного обеспечения, процессоров и программируемых микросхем. Злоумышленник, зная слабости в чем-либо из вышеперечисленного, может создать и отправить по назначению определенный пакет, который вызовет ошибку, переполнение буфера или стека. В результате этого возможны необратимые последствия для всей системы.

Атаки типа «ошибки программирования» условно можно разделить на два типа:

1) Недостатки в программном коде.

Суть атаки заключается в том, что злоумышленник ищет ошибки в программном коде какой-либо программы, либо операционной системы и заставляет ее обрабатывать такие исключительные ситуации, которые она обрабатывать не умеет, за счет чего и возникают ошибки. В качестве примера может служить частая передача пакетов, в которой не учитываются спецификации и стандарты RFC-документов. Злоумышленник наблюдает за тем, справляется ли сетевой стек с обработкой исключительных ситуаций. В случае отрицательного результата передача пакетов приведет прекращению функционирования ядра, либо даже всей системы в целом.

2) Переполнение буфера.

Переполнение буфера возникает в том случае, если программа из-за ошибки программиста записывает данные за пределами буфера. Ярким

примером атаки данного типа может служить ситуация, когда программист написал приложение для обмена данными по сети, которое работает по какому-либо протоколу. В этом протоколе строго указано, что определенное поле пакета максимум может содержать 65536 бит данных. Но после тестирования приложения оказалось, что в ее клиентской части в это поле нет необходимости помещать данные, размер которых не более 255 бит. Поэтому и серверная часть примет не более 255 бит. Далее злоумышленник изменяет код приложения так, что теперь клиентская часть отправляет данные по протоколу 65536 бит, однако сервер к их приему не готов. Из-за этого возникает переполнение буфера, и легальные пользователи не могут получить доступ к приложению[7].

Для предотвращения атак данного типа специалистами в области информационной безопасности рекомендуется тщательное тестирование устанавливаемого программного обеспечения, либо использование только лицензионного программного обеспечения.

1.2.3 Атаки на DNS сервер. Сценарии реализации атак данного типа основываются на том, что злоумышленник, имея доступ к маршрутизатору, изменяет таблицы маршрутизации таким образом, чтобы пользователи, желающие попасть на сервер с одним IP-адресом перенаправлялись на другой, либо несуществующий. Кроме того, имея доступ к промежуточному буферу с быстрым доступом DNS, злоумышленник может привязать искомое доменное имя к совсем другому IP-адресу, и тогда пользователи будут перенаправлены на него.

Атаки на DNS можно условно разделить на две категории:

1) Атаки на уязвимости в DNS-серверах. Результатом проведения атак данного типа является перенаправление пользователя на подставную страницу, с целью получения несанкционированного доступа к его личной информации.

2) DDoS-атаки, приводящие к неработоспособности DNS-сервера. При недоступности DNS-сервера пользователь не сможет попасть на нужную ему страницу, так как его браузер не сможет найти IP-адрес, соответствующий введённому адресу сайта. DDoS-атаки на DNS-сервера могут осуществляться как

за счёт невысокой производительности DNS-сервера, так и за счёт недостаточной ширины канала связи [8].

Основной причиной подверженности DNS-систем угрозам является то, что они работают по протоколу UDP, более уязвимому, чем TCP.

Существует несколько способов атаки на DNS:

1) Создание обманного DNS-сервера. Механизм данной атаки заключается в том, что атакующий ждет DNS-запроса от атакуемого хоста. После того как атакующий получил запрос, он извлекает из перехваченного пакета IP–адрес запрошенного хоста. Затем генерируется пакет, в котором злоумышленник представляется целевым DNS–сервером. Атакуемый принимает атакующего за реальный DNS. Когда клиент отправляет очередной пакет, атакующий меняет в нем IP-адрес отправителя и пересылает далее на DNS. В результате настоящий DNS-сервер считает, что запросы отправляет злоумышленник, а не атакуемый. Таким образом, атакующий становится посредником между клиентом и реальным DNS–сервером. Далее злоумышленник может исправлять запросы клиента по своему усмотрению и отправлять их на реальный DNS. Но перехватить запрос можно, только если атакующая машина находится на пути основного трафика или в сегменте DNS–сервера.

2) Удаленный способ реализации атаки (в случае отсутствия у злоумышленника доступа к трафику клиента). Для генерации ложного ответа необходимо выполнение нескольких условий:

– совпадение IP-адреса отправителя ответа с адресом DNS-сервера;

– совпадение имен, содержащихся в DNS–ответе и запросе;

– DNS–ответ должен посылаться на тот же порт, с которого был отправлен запрос;

– в пакете DNS–ответа поле ID должно совпадать с ID в запросе.

Механизм этой атаки заключается в следующем: клиент посылает на DNS–сервер запрос и переходит в режим ожидания ответа с сервера. Злоумышленник, перехватив запрос, начинает посылать ложные ответные пакеты. В результате на компьютер клиента приходит большое количество ложных ответов, из которых отсеиваются все, кроме одного, в котором совпали ID и порт. Получив нужный

ответ, клиент начинает воспринимать подставной DNS–сервер как настоящий. Атакующий, в свою очередь, в ложном DNS ответе может поставить IP-адрес любого ресурса.

3) Атака непосредственно на DNS–сервера. В результате данной атаки на ложные IP-адреса будут перенаправлены все пользователи, обратившиеся к атакованному DNS. Как и в предыдущем случае, атака может проводиться из любой точки сети. При отправке клиентом запроса на DNS–сервер, последний начинает искать в своем промежуточном буфере с быстрым доступом подобный запрос. Если до атакуемого клиента такой запрос никто не посылал, и он не был занесен в данный буфер, сервер начинает посылать запросы на другие DNS-сервера сети в поисках IP–адреса, соответствующего запрошенному хосту. Для атаки злоумышленник посылает запрос, который заставляет сервер обращаться к другим узлам сети и ждать от них ответа. Отправив запрос, злоумышленник начинает атаковать DNS потоком ложных ответных пакетов. Прослеживается аналогия из предыдущего метода, но в данном случае, атакующему не нужно подбирать порт, так как все сервера DNS обмениваются информацией по выделенному 53 порту. Злоумышленнику остается только подобрать ID. Когда сервер получит ложный ответный пакет с подходящим ID, он начнет воспринимать злоумышленника как DNS и даст клиенту IP–адрес, посланный атакующим компьютером. Далее запрос будет занесен в промежуточный буфер с быстрым доступом, и при последующих подобных запросах пользователи будут переходить на подставной IP-адрес.

Для снижения эффективности атак данного типа экспертами в области сетевой безопасности разработано несколько методов противодействия DNS DoS-атакам:

– создание сети фильтрации. Идея данного средства защиты заключается в том, что маршрутизатор должен принимать пакеты только на том интерфейсе, где доступен адрес источника данного пакета. Реализация данного средства в сети предотвращает прием пакетов с подделанными адресами источника при входе в сеть.

– использование открытых распознавателей. Распознавателем является имя сервера DNS, которое получает и принимает запросы от внешних источников, а затем, либо отвечает на запрос из буферизированных данных, либо пересылает запрос одному или нескольким серверами имен, чтобы получить ответ. Открытые распознаватели позволяют внешним третьим лицам использовать ресурсы организации и скрыть источник исходящего трафика.

– владельцы доменов могут защититься от атак на их родительской зоне, увеличивая TTL, для каждой «делегационной записи». Это записи имен сервера и связанные с ними записи A или AAAA. Следовательно, до тех пор, пока существуют эти записи в промежуточном буфере с быстрым доступом рекурсивного сервера имен, нет никаких оснований для запроса их в родительской зоне. Таким образом, увеличенное время жизни позволяет снизить уязвимость домена к атакам данного типа[4].

– развертывание Domain Name System Security Extensions (DNSSEC) протокола. В основе DNSSEC лежит метод цифровой подписи ответов на запросы. У администратора доменной зоны, поддерживающей данную технологию, есть закрытый ключ, который с помощью криптографических алгоритмов позволяет сгенерировать цифровую подпись. Клиенты же, в свою очередь, получают открытый ключ, соответствующий закрытому. Клиентский ключ дает возможность проверять валидность цифровой подписи [8].

1.2.4. Атаки, приводящие сервер к недостатку ресурсов. Сценарии реализации атак данного типа направлены на исчерпание критических системных ресурсов web-сервера: процессорное время, место, память и т.д.: Злоумышленник отсылает множество запросов на web-сервер, после чего второй перестает обслуживать легальные запросы. Но, в отличие от вышеизложенных типов атак, пакеты не забивают канал хоста атакуемого, а занимают все его процессорное время. Атакующий выбирает такой способ передачи трафика, который позволяет занять у web-сервера много процессорного времени. Каналов может оказаться достаточно, однако процессор может не выдержать такой нагрузки. Результатом данной атаки будет отказ в доступе легальным пользователям к сервисам, предоставляемых web-сервером. Яркими

представителями атак данного типа являются низкоинтенственные DOS-атаки прикладного уровня или Slow HTTP атаки. Особенность реализации таких атак заключается в том, чтобы как можно дольше держать соединение с сервером открытым. Процесс открытия соединения и ожидание ответа почти не требует ресурсов у злоумышленника, однако привязывает процессы сервера к ожиданию запросов. В следствии, сервер будет находиться в режиме ожидания, пока не истечет тайм-аут, а затем закроет соединение.

Открытие нескольких сотен одновременных соединений сделает все доступные процессы сервера занятыми. При достижении максимального количества процессов, сервер зарегистрирует это событие в журнале ошибок и начинает сохранение новых подключений в очередь. Если будут продолжаться открываться новые подключения с высокой скоростью, легальные запросы не смогут быть обслуженными. Если начать открытие соединений с более высокой скоростью, очередь сама по себе переполнится и это приведет к отклонению новых соединений.

1.2.5 Низкоинтенсивные атаки на базе протокола HTTP

Slow HTTP атаки имеют ряд особенностей, которые аргументируют вышеизложенную статистику:

– схожесть атакующих соединений с легальным трафиком;

– традиционные методы обнаружения атак не позволяют обнаруживать атаки данного типа;

– существующие IPS / IDS решения, основанные на сигнатурах, как правило, не распознают атаки данного типа;

– данные атаки требуют мало ресурсов и небольшую полосу пропускания для реализации;

– атаки данного типа выдвигают минимальные требования к производительности атакующего хоста[10].

На основании вышеизложенной статистики и особенностей реализации Slow HTTP атак можно свидетельствовать о необходимости исследования данного типа атак, в целях разработки модели их обнаружения и предотвращения.

1.3. Анализ реализации низкоинтенсивных DOS-атак на базе протокола HTTP

Различные стадии обслуживания запроса могут быть использованы для реализации различных типов Slow HTTP атак. В соответствии с этим, выделяют три разновидности атак низкой интенсивности:

1) Атака низкой интенсивности Slowloris.

Данный инструмент реализации Slow HTTP атак разработан специалистом в области информационной безопасности Робертом Хансеном для проведения атак типа «Denial of Service». Для реализации Slowloris используется уязвимость в архитектуре серверов Apache 1.x, Apache 2.x, dhttpd, GoAhead WebServer и Squid и других популярных web-серверов, которая заключается в ограничении числа одновременно открытых подключений.

Суть атаки заключается в том, что Slowloris заставляет атакуемый web-сервер обслуживать большое количество открытых соединений путем непрерывной отправки незавершенных GET-запросов. В случае, когда все свободные потоки web-сервера будут заняты (количество открытых соединений достигнет максимального значения параметра «Max Clients»), все поступающие запросы будут становиться в очередь, а по истечении тайм-аута отбрасываться, что свидетельствует об отказе в обслуживании легальным пользователям [11].

2) Slow REQUEST BODIES или Slow HTTP POST атаки.

Другой разновидностью атак низкой интенсивности прикладного уровня, является атака Slow Request Bodies или Slow HTTP POST атака, продемонстрированная на OWASP 2010 Application Security Conference исследователем Wong Onn Chee в 2009.

Атака базируется на уязвимости в протоколе HTTP. Slow HTTP POST работает по следующему сценарию: злоумышленник отправляет POST заголовок с легитимным полем «Content-Length», которое позволяет web серверу идентифицировать объём данных, который к нему поступает. Как только заголовок отправлен, тело POST запроса начинает передаваться с очень медленной скоростью, что позволяет использовать ресурсы сервера намного

дольше, чем это необходимо, и, как следствие, не давать web-серверу возможности обрабатывать другие запросы. Несколько тысяч таких соединений могут привести web-сервер к отказу в обслуживании за несколько минут.

Атака приводит к отказу в обслуживании такие web-сервера как Microsoft IIS, Apache и др., работающие в рамках протоколов HTTP или HTTPS и, очевидно, любых «безопасных» подключений вроде SSL, VPN и других. Также атака может быть адаптирована для работы с SMTP и даже DNS-серверами. Более того, это единственная известная науке атака типа отказ в обслуживании, которую реально организовать через прокси-сервер[12].

3) Slow Read атака.

Вышерассмотренные атаки низкой интенсивности, направлены на замедление скорости передачи запросов к web-серверу. Slow Read атака работает по сценарию замедления скорости, с которой клиент (атакующий) способен читать данные ответа на запрос.

Изначально атакующий запрашивает у web-сервера данные (как правило, большого объема), афишируя в запросе размер своего TCP-окна приема. Однако, после установления соединения, злоумышленник изменяет размер TCP-окна на очень маленький для принятия ответных данных. Соответственно, сервер отправляет данные клиенту медленно, сохраняя при этом сокеты открытыми. Он продолжает проверять клиента, чтобы узнать размер его окна приема информации, в то время как клиент постоянно отправляет данные с малым размером TCP-окна, что в результате замедляет передачу. Чем больше размер файла, тем больше времени потребуется для завершения таких соединений. Несколько таких запросов для большого файла могут очень быстро привести сервер к отказу в обслуживании[13].

1.3.1 Реализация Slow HTTP атак

В августе 2011 года Сергей Shekyan написал инструмент «slow http test», который тестирует web-сервера на наличие уязвимостей, связанных с обработкой низкоинтенсивных HTTP запросов, таких как Slowloris и Slow POST и Slow READ. При помощи данного инструмента, на web-сервер Apach 2 были реализованы описанные выше атаки.

Сценарий реализации Slowloris атаки:

Клиент-злоумышленник посылает запрос частями очень маленького объема (13 байт), с задержкой между ними в 5 сек, что позволяет ему долго держать соединение открытым (рис. 1.2):

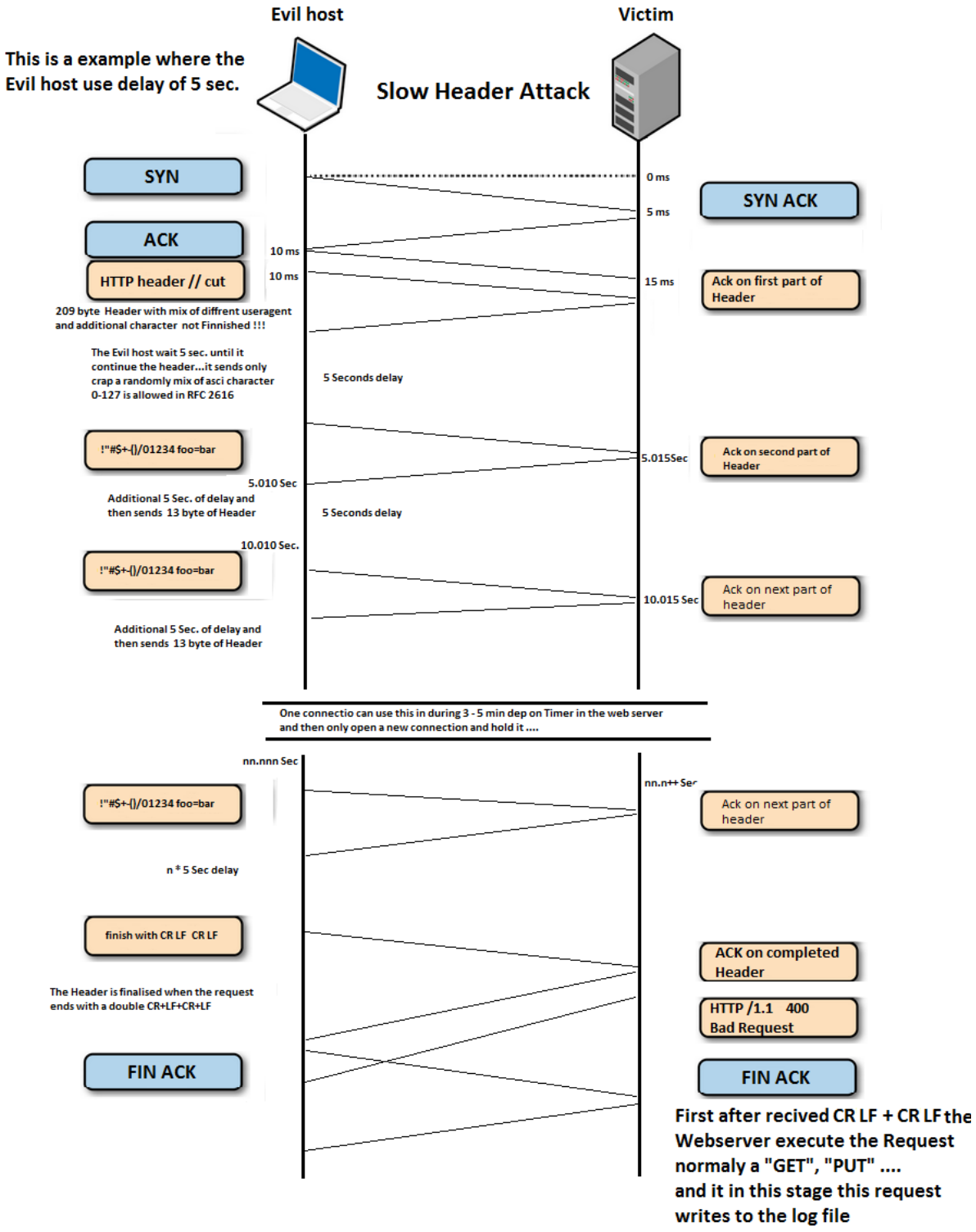


Рисунок 1.2. Сценарий реализации Slowloris атаки

В конце запроса клиент должен послать серверу окончательное CRLF, которое сообщает ему, что запрос завершен. Если этого сделано не будет, web-сервер будет ожидать остальные данные запроса, пока не достигнет интервала ожидания. Таким образом, Slowloris может держать web-сервер в постоянном режиме ожидания, посылая новые запросы, до момента достижения тайм-аута.

Сценарий реализации Slow POST атаки:

Атакующий посылает тело запроса частями очень маленького объема (15 байт), с задержкой между ними в 10 сек, что позволяет ему долго держать соединение открытым (рис.1.3).

В сценарии реализации данного типа атаки, изначально необходимо создать нормальное тестовое соединение с сервером, в целях получения информации о таймауте соединений web-сервера. Было выяснено, что для данной реализации тайм-аут сервера равняется 10 секунд. Соответственно, одно соединение будет длиться не более 10сек.

Сценарий реализации HTTP Slow READ атаки:

Клиент-злоумышленник посылает запрос серверу, указывая размер TCP-окна, для приема информации в 572 байта. Однако, когда сервер начинает посылать данные клиенту, клиент сообщает о том, что размер его окна для приема данных равен 0.

Соответственно сервер постоянно пытается отправить данные клиенту, однако, клиент, читает их «очень медленно», поддерживая соединение при помощи функции web-сервера «Keep alife», что позволяет постоянно удерживать соединение открытым (рис.1.4).

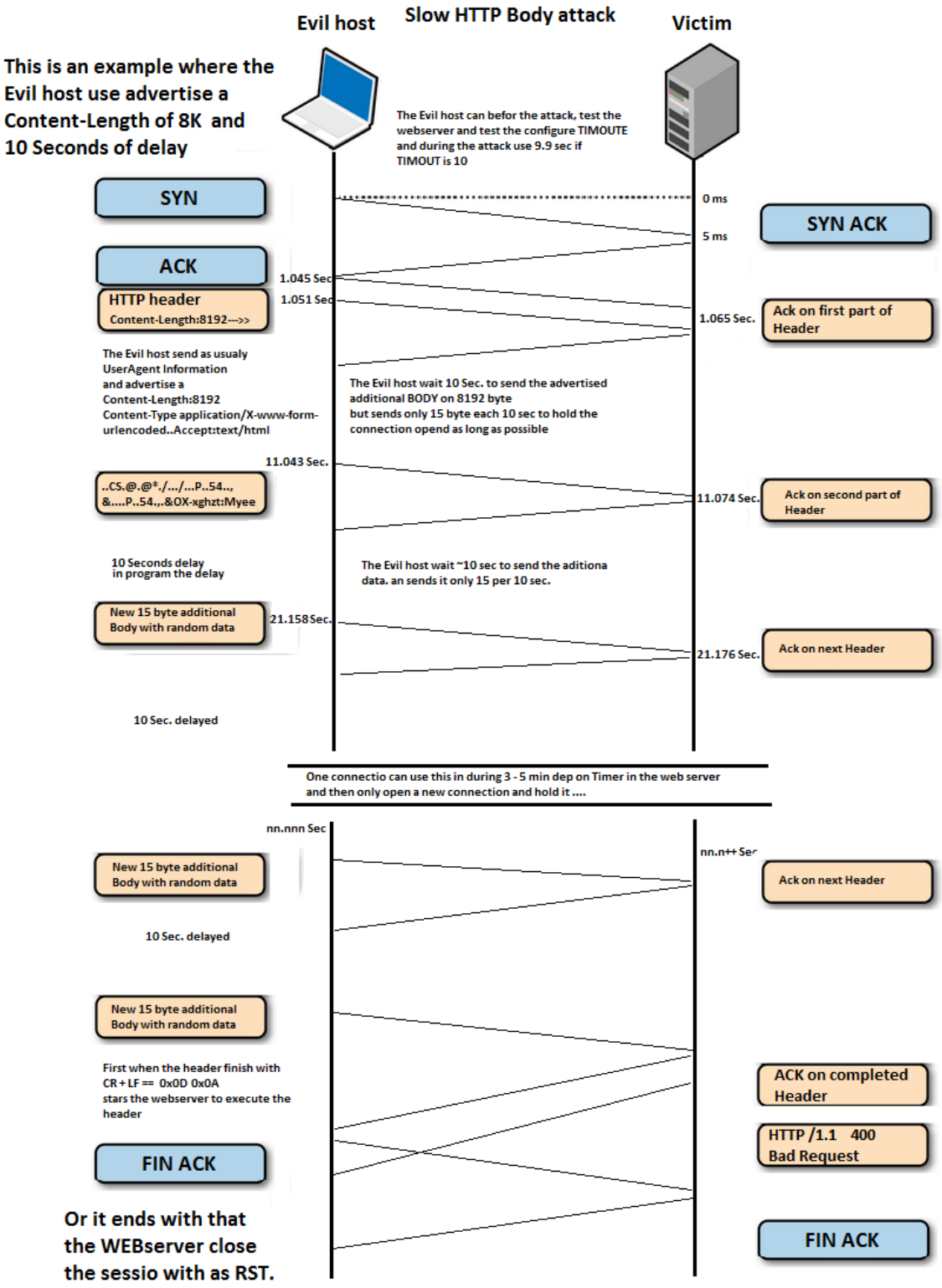


Рисунок 1.3. Сценарий реализации HTTP Slow Body атаки

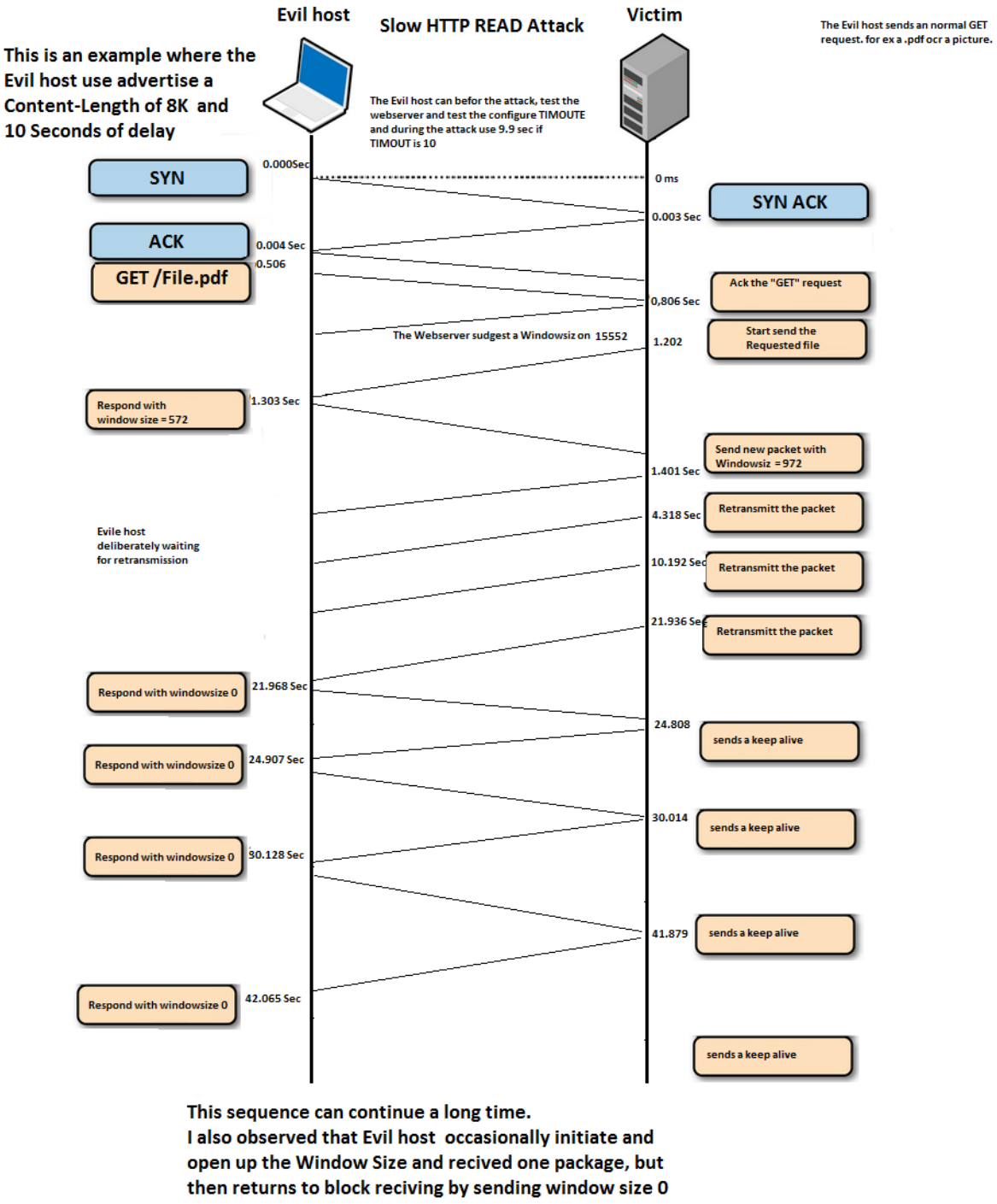


Рисунок 1.4. Сценарий реализации HTTP Slow READ атаки

Таким образом, при помощи инструмента «slow http test» было реализовано три типа низкоинтенсивных прикладных атак: Slowloris, Slow HTTP POST и Slow READ. Первые две атаки направлены на передачу HTTP-запроса с длительной задержкой между передачей частей запроса: в первом случае – медленная передача заголовка запроса, во втором случае – тела запроса. Третий тип атак медленно читает ответ сервера. Изначально Slow READ афиширует

web-серверу большой размер TCP-окна приема, и, далее, меняет размер окна на очень маленький, что заставляет web-сервер постоянно находиться в режиме ожидания передачи данных. Все три типа атак, позволяют держать соединение с сервером открытым в течение длительного времени, позволяя им исчерпать все ресурсы web-сервера, не разрешая легальным пользователям получить услуги.

Атаки данного типа в наше время представляют собой угрозу для web-серверов, так как со стороны атакуемого диагностика и обнаружение данного типа атак составляет сложность для администраторов трафик не превышает нормальных значений. Схожесть атакуемого трафика с легальным усложняет фильтрацию атакующих пакетов, что позволяет злоумышленнику достаточно быстро и легко добиться отказа в обслуживании web-сервера.

Большинство существующих систем обнаружения вторжений направлены на обнаружение и предотвращение атак сетевого и транспортного уровня, суть которых заключается в генерации значительного трафика, позволяющего заполнить всю пропускную способность атакуемого узла. Особенностью Slow HTTP атак является отсутствие больших объемов трафика. Таким образом, средства направленные на обнаружение и предотвращение DoS-атак сетевого и транспортного уровня в данном случае показывают низкую эффективность.

Следовательно, возникает необходимость разработки математической модели, которая позволит формализовать атаки данного типа, а так же разработать алгоритм их обнаружения.

1.4. Анализ средств обнаружения DOS-атак

Анализ статистики компаний, предоставляющих услуги сетевой защиты показал, что за последние несколько лет наблюдается значительный рост объемов и сложности DOS-атак. В тоже время традиционные средства сетевой защиты (файерволы и системы обнаружения вторжений) не позволяют эффективно с ними бороться.

Специализированные средства защиты от DOS-атак основывают свою работу на анализе трафика и обнаружении аномалий в его структуре. В соответствии с таким подходом выполняется построение характеристик трафика в штатном режиме работы сети и поиск аномалий на наблюдаемом интервале. В

данном случае аномалия трафика – это отклонение характеристик от статистических значений собранных профилей. Отклонение больше заданного порога – вызывает срабатывание сигнала тревоги.

1.4.1 Формализация параметров низкоинтенсивных атак

Для обнаружения Slow-атак прикладного уровня необходим постоянный мониторинг систем в режиме реального времени, с составлением отчетов по распределению ресурсов. Ведя непрерывный анализ, защитный инструмент, поддерживающий учет ресурсов, должен обнаруживать отклонения от нормы значений интенсивностей поступления и обработки заявок на web-сервере на основе входных данных, характеризующих поведение web-сервера.

Параметры сервера, в случае возникновения каждого типа низкоинтенсивных атак представлены в таблице 1.2.

Анализ совокупности значений этих условий позволяет формализовать набор параметров, характеризующих состояние web-сервера, необходимых для осуществления Slow HTTP атак:

Для Slowloris и Slow POST характерными параметрами являются:

1) Количество запросов.
2) Количество ip-адресов.
3) Скорость соединения.
4) Интервал активности клиента близкий к тайм-ауту.
5) Соотношение заявленного размера окна и передаваемых данных.

Таблица 1.2

Параметры web-сервера, в случае возникновения Slow HTTP атак

Slowloris	Slow POST	Slow READ
Одновременное количество запросов с одного IP-адреса (варьирует от нескольких сотен до нескольких десятков тысяч запросов, в зависимости от масштабности сервера;		Включены постоянные соединения (Keep-Alive) и HTTP конвейер.
Значение задержки между передачами частей запроса стремится к величине тайм-аута соединения сервера, но не достигает его;		Значение задержки получения данных ответа сервера стремится к величине тайм-аута соединения сервера, но не достигает его;

Отправка заголовка запроса маленькими частями (10-20байт);	Отправка тела запроса маленькими порциями (10-20байт);	Первоначальный размер окна приема достаточно большой;
Ожидание двойного CRLF практически до достижения значения тайм-аута.	Достаточно большое поле «content length», с сравнительно маленькими блоками передаваемых данных.	На web-сервер приходят SYN-пакеты с аномально малым размером TCP окна.

Для Slow READ характерными параметрами являются:

1) Количество запросов.

2) Количество ip-адресов.

3) Скорость соединения.

4) Интервал активности клиента близкий к тайм-ауту.

5) Разница между первоначальным и последующим размерами TCP окна (стремится к 0).

6) Наличие или отсутствие постоянных соединений (Keep-Alive) и HTTP конвейер.

Данная формализация необходима для сравнительного анализа существующих подходов к обнаружению вторжений. Выделенные параметры выступают в качестве критериев атаки, и в зависимости от получаемого ими значения, могут свидетельствовать о наличии атаки. В зависимости от алгоритма обнаружения и математического аппарата, используемого в системе обнаружения атак, данные критерии могут выступать в качестве входных данных в сценариях обнаружения, либо нести за собой характер наличия либо отсутствия атаки.

1.4.2. Формальная постановка задачи метода обнаружения Slow HTTP атак. Основываясь на признаки, характерные Slow HTTP атакам, и на специфический характер их реализации, описанный в 1.3, задачу обнаружения можно сформулировать как: обнаружение момента начала атаки и предотвращение атаки, до момента отказа в обслуживании web-сервера:

Пусть задано:

Последовательность наблюдаемых состояний web-сервера:

$$S = S1, S2, S3, , \dots \dots Sn. \quad (1.1)$$

Время перехода web-сервера в каждое из состояний:

$$T = T1, T2, T3, , \dots \dots Tn. \quad (1.2)$$

Множество совокупностей значений параметров сервера, свидетельствующих о наличии атаки и являющимися пороговыми значениями:

$$X = \{X_1^i, X_2^i, \dots . . X_n^i\}, \quad (1.3)$$

где:

$i -$ тип Slow HTTP- атаки.

Требуется оценить вероятность и время перехода web-сервера в состояние «отказа в обслуживании».

Для выполнения поставленной задачи необходимо разработать метод, который позволит обнаруживать момент начала атаки и предотвратить атаку, до момента отказа в обслуживании web-сервера.

1.5. Выводы по разделу.

1. В ходе анализа мультисервисных сетей, использующих облачную инфраструктуру в качестве технологической платформы определено, что такие системы в значительно мере подвержены атакам на отказ в обслуживании (DOS).

2. Особенностью DOS атак на облачные инфрастуруктуры является то, что они в значительной мере ориентированны на прикладной уровень. Данный факт приводит к тому, что сущесвтующие системы обнаружения DOS атак показывают низкую эффективность.

3. Существующие системы обнаружения основанны на поиске аномалий в траффике, в тоже время DOS атаки, реализуемые на прикладном уровне не

генерируют большого объема траффика и, следовательно, не вызывают аномалий в привычном их понимании.

4. В разделе рассмотрены особенности реализации DOS-атак прикладного уровня (Slow-HTTP) примере web-сервера «Apatch 2». Реализовано три типа низкоинтенсивных прикладного уровня: Slowloris, Slow HTTP Post и Slow Read.

5. Выделены основные параметры соединения в момент отказа в обслуживании для всех типов Slow HTTP атак: Количество сессий, за заданный интервал времени, тип запроса, значение поля заголовка Content-Length, объем данных по каждой сессии, задержка между пакетами внутри сессии, скорость соединения, продолжительность тестирования, длительность открытых соединений, полученный диапазон окна, количество одновременных запросов в одном соединении, скорость чтения буфера приема.

6. Проведен сравнительный анализ существующих подходов к обнаружению Dos - атак. Выделены как достоинства, так и недостатки каждого метода, применительно к обнаружению Slow HTTP атак. Принято решение о необходимости разработки нового метода обнарушения DOS атак, ориентированного на обнаружение атак прикладного уровня.

7. Выполнена формальная постановка задачи на обнаружение низкоинтенсивной DOS-атаки на web-сервер.

РАЗДЕЛ 2 РАЗРАБОТКА МАТЕМАТИЧЕСКОЙ МОДЕЛИ ОБНАРУЖЕНИЯ SLOW-HTTP АТАК

2.1 Анализ существующих подходов к обнаружению DOS атак на облачную инфраструктуру.

В данный момент на практике используется несколько математических аппаратов, которые позволяют разработать методы обнаружения DOS атак на облачную инфраструктуру. Такими средствами являются: поход на основе конечных автоматов, детерминированный поход на основе правил, подход на основе правил нечеткой логики, подход, на основе нейронных сетей, подход на основанный на генной инженерии подход, на основе Марковских цепей.

2.1.1 Поход на основе конечных автоматов.

Обнаружение атак с использованием похода, на основе конечных автоматов (КА) основано на моделировании процессов информационного взаимодействия клиентов с web-сервером по протоколам передачи данных. Конечный автомат (рис 2.1) описывается множествами входных данных, выходных данных и внутренних состояний.

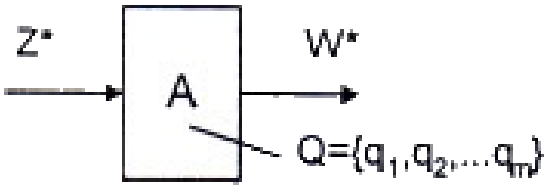


Рис. 2.1 Процесс работы конечного автомата

Формальное определение КА в терминах дискретной математики и теории множеств описывается шестью компонентами:

$$A = \{ I, O, S, S0, P, D \}; \quad (2.1)$$

– множество входных данных web-сервера:

$$I = \{Ii\}; \quad (2.2)$$

– множество выходных данных web-сервера

$$O = \{Oj\}; \quad (2.3)$$

– множество состояний web-сервера

$$S = \{Sk\}; \quad (2.4)$$

– начальное состояние web-сервера $S0$;

– функция переходов

$$P(S, I) \ => S, \quad (2.5)$$

которая ставит в соответствие каждой паре «состояние – входные данные» новое состояние;

– функция выходов

$$D(S, I) \ => O, \quad (2.6)$$

которая ставит в соответствие каждой паре «состояние – выходные данные» выходные данные.

Если поведение web-сервера описывается набором правил вида: «Находясь в состоянии A, при получении данных S web-сервер переходит в состояние B и при этом выполняет действие D», то такая система будет представлять собой конечный автомат (рис. 2.2):

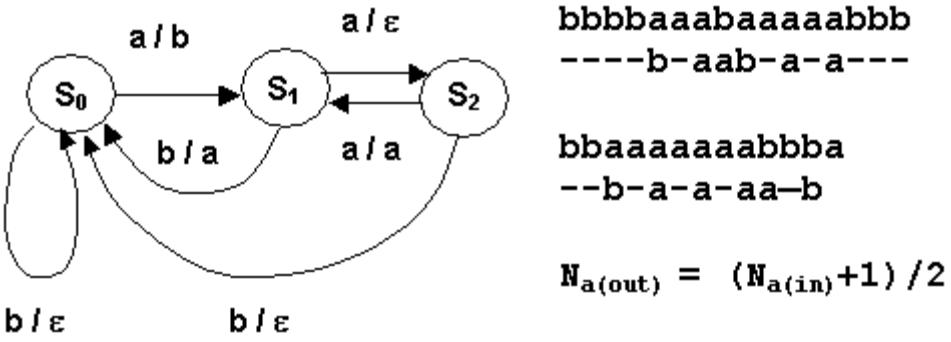


Рис. 2.2. Диаграмма состояний-переходов конечного автомата

.

Получая входные цепочки, содержащие пакеты с данными типа a и b в произвольном порядке, КА формирует аналогичные цепочки данных более регулярного вида. Так, из подряд идущих пакетов типа b остается один, а в последовательности типаa остаются все нечетные пакеты, кроме первого. При нечетном количестве данных типаa на входе КА добавляет еще один a в выходную последовательность. Отсюда вычисляется закон преобразования длины входной цепочки данных типа a:

$$Na(out) = (Na(in) + 1)/2. \quad (2.7)$$

Анализируя состояния и условия переходов в них, можно определить их содержательный «смысл»: состояния $S1$,$S2$ обозначают нечетное и четное число подряд прочитанных данных типа a.

Атаки фиксируются по переходам системы из состояния в состояние. Предполагается, что в системе «штатные» переходы из состояния в состояние определены, а неизвестные состояния и переходы в эти состояния регистрируются как аномальные. Достоинством этой модели является упрощенный подбор классификационных признаков для системы и рассмотрение малого числа переходов из состояния в состояние. Модель позволяет обнаруживать атаки в потоке обработки данных сетевыми протоколами в режиме близком к реальному масштабу времени. К недостаткам модели следует отнести необходимость разработки большого количества

сложных экспертных правил для сравнительного анализа состояний web-сервера [14].

2.1.2 Детерминированные подходы на основе правил

Данный подход используется для обнаружения Slow HTTP атак, опираясь на сравнительный анализ характеристик web-сервера, как при нормальной загрузке, так и в случае атаки.

Изначально для всех субъектов анализируемой системы определяются профили. Любое отклонение (т. е. дисперсия) используемого профиля от эталонного считается несанкционированной деятельностью. Средние частоты и величины переменных вычисляются для каждого типа нормального поведения:

$T1$-количество одновременных соединений с одного IP-адреса;

$T2$- среднее время поступления запроса;

$T3$-среднее время обслуживания запроса;

$T4$- соотношение заявленного размера окна и передаваемых данных.

Составляется таблица 2.2 корреляций Rij, между параметрами сервера

Таблица 2.1

Коэффициенты корреляции параметров web-сервера

	T1	T2	T3	T4
T1	R_{11}	R_{12}	R_{13}	R_{14}
T2	R_{21}	R_{22}	R_{23}	R_{24}
T3	R_{31}	R_{32}	R_{33}	R_{34}
T4	R_{41}	R_{42}	R_{43}	R_{44}

Далее выделяются связанные между собой требования к web-серверу, которые и являются основой профилей (факторов). Составляется таблица 2.2 факторов, каждому из которых будет соответствовать среднее значение коэффициента корреляции соответствующих переменных по фактору.

Таблица 2.2

Факторная матрица коэффициентов корреляции параметров web-сервера

	Профиль A	Профиль B	Профиль N
T1	$<R_{1A}>$	$<R_{1B}>$	$<R_{1N}>$
T2	$<R_{2A}>$	$<R_{2B}>$	$<R_{2N}>$
T3	$<R_{3A}>$	$<R_{3B}>$	$<R_{3N}>$
T4	$<R_{4A}>$	$<R_{4B}>$	$<R_{4N}>$

О возможности атаки сообщается, когда наблюдаемые значения профилей выпадают из нормального диапазона, т. е. превышают заданный порог.

Важной задачей является правильный выбор контролируемых параметров для системы обнаружения атак. Малое их число или неправильно отобранные параметры могут привести к тому, что модель описания поведения web-сервера будет неполной, и атаки останутся за пределами ее рассмотрения. С другой стороны, слишком большое число параметров мониторинга вызовет снижение производительности контролируемого узла за счет увеличенных требований к потребляемым ресурсам (оперативной и дисковой памяти, загрузке процессора и т. д.).

К достоинствам данного подхода можно отнести их способность обнаруживать неизвестные атаки и способность адаптироваться к изменению поведения клиента.

И хотя данный подход достаточно эффективен и надежен для некоторых типов атак, применения в обнаружении Slow HTTP атак он не получил из-за своих недостатков:

1) В данном подходе вероятность получения ложных сообщений об атаке является гораздо более высокой, чем при использовании других подходов;

2) Статистический подход на основе правил не корректно обрабатывает изменения в деятельности. В результате могут появиться как ложные сообщения об опасности, так и ложные отрицательные сообщения (пропущенные атаки);

3) Статистический подход не способен обнаружить атаки со стороны субъектов, для которых невозможно описать шаблон типичного поведения;

4) Статистический подход не позволяет обнаруживать атаки со стороны субъектов, которые с самого начала выполняют несанкционированные действия. Таким образом, шаблон обычного поведения для него будет включать только атаки;

5) Данный подход должен быть предварительно настроен (заданы пороговые значения для каждого параметра для каждого пользователя);

6) Данный подход нечувствителен к порядку следования событий.

Следует отметить, что преимуществом использования этого подхода является то, что системы, которые его используют, не требуют постоянного обновления базы сигнатур атак, что позволяет не тратить большие деньги на поддержание команды разработчиков, постоянно обновляющих систему обнаружения атак [15].

2.1.3 Подход на основе экспертной оценки

Экспертные системы состоят из набора правил, которые охватывают знания человека-эксперта. Использование экспертных систем представляет собой распространенный метод обнаружения атак, при котором информация об атаках формулируется в виде правил. Эти правила могут быть записаны, например, в виде последовательности действий или в виде сигнатуры. При выполнении любого из этих правил принимается решение о наличии несанкционированной деятельности. Важным достоинством такого подхода является практически полное отсутствие ложных тревог.

База данных (БД) экспертной системы должна содержать сценарии большинства известных на сегодняшний день атак. Для того чтобы оставаться постоянно актуальными, экспертные системы требуют постоянного обновления БД. Хотя экспертные системы предлагают хорошую возможность для просмотра данных в журналах регистрации, требуемые обновления могут либо игнорироваться, либо выполняться администратором вручную. Как минимум, это приводит к экспертной системе с ослабленными возможностями. В худшем случае отсутствие надлежащего мониторинга снижает степень защищенности

всей сети, вводя ее пользователей в заблуждение относительно действительного уровня защищенности.

Несмотря на то, что в большинстве случаев экспертные системы применяются для анализа трафика, есть методы, которые используют правила для обнаружения аномалий. Метод прогнозируемых порождаемых шаблонов заключается в том, что будущие события предсказываются на основе уже случившихся. Такое правило может быть записано в следующем виде:

$$П_1 - П_2 \Rightarrow (П_3 = N\%, П_4 = M\%, П_5 = (100 - N - M)\%). \quad (2.8)$$

Применительно к обнаружению Slow HTTP атак, данное правило аргументируется таким образом: событие $П_2$ характеризует переход Web-сервера в состояние $S_{2.;}$ данное событие произошло после события $П_1$, что соответствует переходу web-сервера из состояния S_1 в $S_{2,}$и с вероятностью N% следующим событием будет событие $П_3$, с вероятностью M% – $П_4$и с вероятностью (100-N-M)% – событие $П_5$, что соответствует переходу web-сервера в состояние $S_{5.}$

Основным недостатком данного подхода является невозможность обнаружения неизвестных атак. При этом даже небольшое изменение уже известной атаки может стать серьезным препятствием для функционирования системы обнаружения атак. Однако, к преимуществам данного подхода можно отнести простоту его реализации, достаточно высокую скорость функционирования и высокий уровень отсутствия ложных тревог.

Современные коммерческие системы обнаружения атак используют в своей работе примерно следующее соотношение статистических и экспертных методов – 30% и 70%. К сожалению, экспертные системы нуждаются в постоянном обновлении для того, чтобы оставаться актуальными. Системы на основе правил не способны обнаруживать сценарии атак, которые происходят в течение продолжительных периодов времени. Любое разделение атаки либо с течением времени, либо среди нескольких, по - видимому не связанных между собой злоумышленников, также привносит трудности для обнаружения при помощи этих методов [15].

2.1.4 Подход на основе правил нечеткой логики

Подход на основе правил нечеткой логики позволяет описывать правила в незавершенном режиме, в котором последние основываются на знаниях и весах событий, позволяющих предположить вероятность несанкционированного вторжения в сеть.

Процесс получения нечетких заключений соединяет в себе все основные концепции теории нечетких множеств: функции принадлежности, лингвистические переменные, методы нечеткой импликации и т.п.

База правил систем нечеткого вывода предназначена для формального представления эмпирических знаний в форме нечетких продукционных правил. Правила строятся на основании степени принадлежности элементов множеству. Для обнаружения Slow HTTP атак строятся нечеткие правила соответствия параметров входящего трафика эталонному (нормальному) трафику, и строятся функции принадлежности значений данных параметров множеству значений параметров запросов. Чем выше вероятность принадлежности совокупности данных значений множества эталону, тем выше вероятность обнаружения этих атак.

Основой любой нечеткой системы обнаружения сетевых атак являются три характерных структурных компонента, реализующие процесс нечеткого вывода (рис. 2.3):

– фаззификатор;
– аппарат нечеткого вывода;
– дефаззификатор.

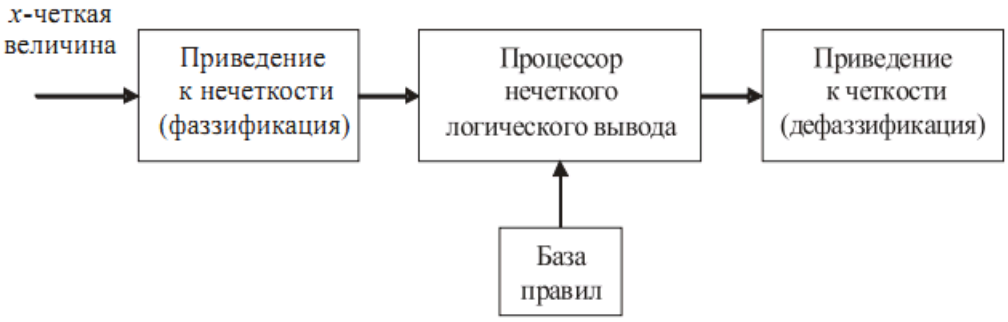


Рис. 2.3. Механизм нечеткого логического вывода

Фаззификатор предназначен для получения формальных нечетких оценок параметров сетевого трафика по устным суждениям эксперта. Поскольку процесс фаззификации в общем случае не формализуем, то роль первого структурного блока часто играет человек.

Аппарат нечеткого вывода осуществляет процесс анализа формальных нечетких оценок с целью выявления отклонений одного и более параметров от заданных норм. В случае обнаружения такого отклонения аппарат нечеткого вывода передает результаты анализа на следующий структурный компонент – дефаззификатор. Этот компонент осуществляет процесс, обратный к процессу фаззификации, т.е. по нечетким результатам анализа вычисляет точечные числовые оценки ситуации (вероятность реализации атаки).

Дефаззификация нечеткого множества:

$$C = \int_{[\underline{x};\bar{x}]} \mu_A(x) / x \tag{2.9}$$

по методу центра тяжести осуществляется по формуле:

$$c = \frac{\int_{\underline{x}}^{\bar{x}} x \times \mu_A(x) dx}{\int_{\underline{x}}^{\bar{x}} \mu_A(x) dx}, \tag{2.10}$$

где:

$\mu_A(x)$ - функция принадлежности элементов множества x нечеткому множеству А.

Несмотря на достоинства данного подхода, такие как быстрое и эффективное обнаружение известных сценариев атак, а так же оперативное создание сигнатур в базе, применительно к обнаружению Slow HTTP атак, подход на основе нечеткой логики обладает рядом недостатков:

– граница между "нормальными" и "аномальными" процессами размыта, что вызывает ряд проблем в реализации обнаружения;

– склонность к «ложным тревогам»;

– склонность пропустить аномальный процесс [16].

2.1.5 Подход, на основе нейронных сетей

Подход на основе нейронный сетей основывается на том, что нейронная сеть проводит анализ информации и предоставляет возможность оценить, что данные согласуются с характеристиками, которые она научена распознавать. В то время как степень соответствия нейросетевого представления может достигать 100%, достоверность выбора полностью зависит от качества системы в анализе примеров поставленной задачи.

Построение нейронной сети позволяет идентифицировать нормальное поведение системы по функции распределения получения пакетов, обучить нейронную сеть и произвести сравнительный анализ событий по обучающей выборке.

Структура нейронной сети представляет собой аналог байесовского классификатора, восстанавливающего распределение вероятностей предъявляемых классов образов. Предложенная архитектура представлена на рисунке (рис. 2.4):

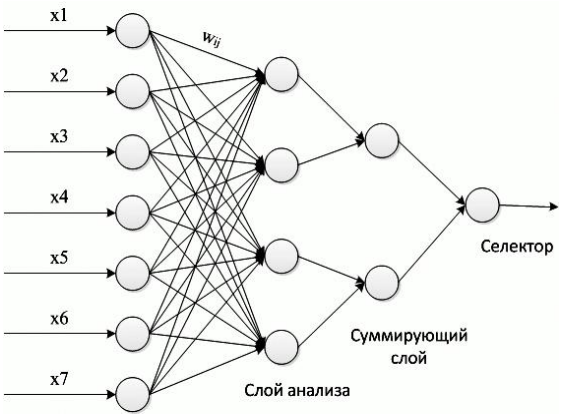


Рис. 2.4. Архитектура модели нейронной сети

$x1 \ldots x7$ – нормированные значения атак, полученные статистически.

Число нейронов слоя анализа образов соответствует числу самих образов. W – весовая матрица, значения которой выбираются и устанавливаются в режиме обучения сети. Если xlпринадлежит Ci(то есть, элемент множества образов принадлежит классу Ci), то по окончании режима обучения wi будет равняться xl. Поскольку векторx нормализирован, векторw также будет нормализированным. После этого устанавливается связь между l-м нейроном слоя анализа образов al и j-м нейроном суммирующего слоя Sj.

Процесс обучения сети состоит в присвоении соответствующих значений входных сигналов элементам весовой матрицы W и установлению связей между нейронами слоя анализа образов и суммирующего слоя.

В режиме классификации на вход сети подается сигнал x и производится активация нейронов слоя анализа образов. При этом находится взвешенная сумма:

$$z_j = \sum_{i=1}^{N} x_i \, w_{ij}. \qquad (2.11)$$

При активации нейрона слоя анализа образов вычисляется величина:

$$f(z_j) = exp\left\{\frac{z_j - 1}{\sigma^2}\right\}. \qquad (2.12)$$

Число нейронов суммирующего слоя соответствует числу различаемых классов.

Выходной нейрон представляет собой селектор, выбирающий нейрон суммирующего слоя с максимальным значением и относящий его к соответствующему классу.

Для корректного обучения сети необходимо предоставить для обучения достаточное количество примеров, полученных в практических условиях, при исследовании конкретной сети, учитывая априорные вероятности возникновения атак.

Преимуществом в использовании нейросетей при обнаружении злоупотреблений является гибкость, которую эти сети предоставляют. Нейросеть способна анализировать данные от сети, даже если эти данные

являются неполными или искаженными, обладает возможностью проводить анализ данных в нелинейном режиме. Более того, способность обрабатывать данные от большого количества источников в нелинейном режиме является особенно важной, поскольку некоторые атаки могут быть проведены против сети скоординировано многочисленными хакерами (DDOS-атаки). Так как защита вычислительных ресурсов требует своевременной (быстрой) идентификации атак, скорость обработки в нейросети может быть достаточной для реагирования в реальном времени на проводимые атаки до того, как в системе появятся непоправимые повреждения целостности системы, что является еще одним достоинством этого подхода.

Выходные данные нейронной сети выражаются в форме вероятности, т. е. нейронная сеть предоставляет возможность прогнозирования дальнейших атак. Система обнаружения атак на основе нейронной сети идентифицирует вероятность того, что отдельное событие, либо серия событий указывают, что против системы осуществляется атака. По мере того, как нейросеть "набирается опыта", она будет улучшать свою способность определять, в каком из этих событий наиболее вероятно имеются признаки атаки. Затем собранная информация может быть использована для генерации серии событий, которые должны реализовываться, если бы действительно имела место попытка атаки. Отслеживая последовательное местонахождение этих событий, система способна улучшить анализ событий и провести защитные мероприятия прежде, чем атака станет успешной. Кроме того, механизм обучения позволяет не тратить средства на длительные исследования новых атак и алгоритмов их обнаружения [15].

Однако наиболее важное преимущество нейросетей при обнаружении атак заключается в способности нейросетей "изучать" характеристики умышленных атак и идентифицировать элементы, которые не похожи на те, что наблюдались в сети прежде.

Основным недостатком применения нейросетей для детектирования вторжения является природа "черного ящика" нейросети. Он может принять

решение об отнесении события к классу атак, но не объясняет, почему такое решение было принято [17].

2.1.6. Подход, основанный на генной инженерии

Подход к обнаружению атак на основе генной инженерии опирается на применение в сфере информационных технологий достижений генетики и моделей иммунной системы человека. Подход этой модели базируется на моделировании элементов иммунной системы человека в средствах обнаружения атак путем представления данных о технологических процессах в системе цепочкой (вектором) признаков и затем вычисления меры сходства между обучающей цепочкой признаков, характеризующих нормальное «поведение» системы и тестовой цепочкой, характеризующей ее аномальное функционирование. Если согласование между данными обучающей и тестовой цепочек не найдено, то процесс интерпретируется как аномальный.

Одна из основных трудностей применения этой модели состоит в выборе порога согласования данных, формирования необходимого объема данных обучающей и тестовой выборки и чувствительности к ложным срабатываниям.

Недостаток данного подхода заключается в том, что требуется сложная процедура настройки обучающей и тестовой выборок, или данных о поведении индивидуума в системе с привлечением высококвалифицированного оператора [18].

2.1.7 Подход, на основе Марковских цепей

Подход к обнаружению атак на основе цепей Маркова заключается в описании системы в виде состояний, изменяющихся с течением времени. Если состояние системы меняется во времени случайным, заранее непредсказуемым образом, подразумевается, что в системе протекает случайный процесс. Если для каждого момента времени вероятность любого состояния системы в будущем зависит только от ее состояния в настоящем и не зависит от того, когда и каким образом система пришла в это состояние, то такой процесс называется «Марковским». Нахождение системы в определенном состоянии или переход в соседнее, зависит от изменения параметров данной системы с течением времени. Данный подход позволяет спрогнозировать нахождение системы в

определенном состоянии при помощи вероятностного графа состояний (рис. 2.5) и матрицы переходов системы из одного состояния в другое, что позволяет оценить вероятность осуществление атаки на информационную систему.

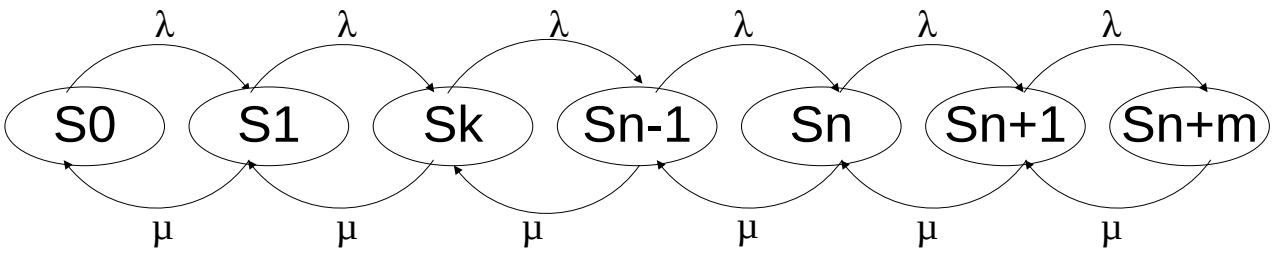


Рис. 2.5. Пример графа состояний

Вероятности нахождения системы в определенном состоянии рассчитываются при помощи уравнений, полученных из матрицы переходов состояний и уравнений Эрланга:

$$\begin{cases} -\lambda p_0 + \mu p_1 = 0 \\ \lambda p_0 - (\lambda + \mu) p_1 + 2\mu p_2 = 0 \\ при \quad (0 < k < n) \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ \lambda p_{n-1} - (\lambda + n\mu) p_n + n\mu p_{n+1} = 0 \\ \lambda p_{n+s-1} - (\lambda + n\mu) p_{n+s} + n\mu p_{n+s+1} = 0 \\ при \quad (1 \le S \le m) \\ \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots \\ \lambda p_{n+m-1} - n\mu p_{n+m} = 0 \end{cases}, \quad (2.12)$$

при

$$\sum_{k=0}^{n+m} p_k = 1, \quad (2.13)$$

где:

λ-плотность потока поступления заявок;

μ-плотность потока обслуживания заявок;

p_0-вероятность нахождения сервера в начальном состоянии;

s-количество обслуженных заявок;

m-размер очереди;

n-общее количество каналов;

k-количество занятых каналов.

Данный подход позволяет обнаруживать атаки на систему, на основе входного потока данных о системе, однако его недостатком является отсутствие возможности отслеживать ситуацию в системе в реальном режиме времени, что снижает его эффективность к процессу обнаружения атак [19].

2.2. Описание поведения сервиса через набор состояний

Вышеизложенные методы позволяют эффективно распознавать и бороться с лавинообразными DDoS-атаками сетевого и транспортного уровня, направленными на заполнение пропускной способности каналов (Smurf, UDP-флуд и др.) и превышение нормальной загрузки отдельных узлов сети (SYN-флуд, Teardrop, Ping of death и др.).

В тоже время указанные подходы малоэффективны для обнаружения низкоинтенсивных DOS-атак прикладного уровня, особенностью которых является отсутствие аномалий в характеристиках трафика. Отличить трафик, генерируемый в ходе таких атак от легального трафика достаточно сложно. Следовательно, применение сигнатурных методов так же является не эффективным. Указанный класс атак появился сравнительно недавно, но как показывает статистика, доля низкоинтенсивных атак растет из года в год. Как правило, низкоинтенсивные атаки приводят к отказу web-серверов, но в тоже время они могут быть адаптированы для воздействия на любую систему прикладного уровня.

Основываясь на анализ недостатков вышеизложенных подходов, очевидным фактом является невозможность выделения математического аппарата, на основе которого можно построить эффективную модель обнаружения Slow HTTP атак.

Соответственно, модификация вышеизложенных подходов и использование их в дополнении с другими существующими математическими аппаратами дает возможность построения модели, которая позволит обнаруживать и прогнозировать возникновение Slow HTTP атак, на основании выделенных параметров web-сервера.

Для описания рабочих процессов в облачной инфраструктуре предлагает использовать теоретико-множественный подход. В соответствии с данным подходом процесс обработки информации представляется набором состояний объекта обработки.

Обслуживание запросса некоторым сервисом подразумевает выполнение последовательности элементарных операций – открыть доступ по операции, выполнить операцию, закрыть доступ по операции. Данный процесс соответствует схеме работы TCP простокола (открыть соединение, передать данные, закрыть соединение). В ходе реализции доступа элементарные операции выполняются мгновенно, но могут отстоять друг от друга во времени. Для каждого объекта также определен конечный набор прав доступа – правила, определяющие, к каким типам объектов он может иметь доступ. Любой доступ к объекту, не соответствующий этим правилам, будет иметь признак атаки. Задача систем систем сетевой безопасности – обнаружение таких ситуаций и их обработка.

Обнаружение атаки может происходить в процессе ее реализации или после ее завершения. Каждый объект (сервис) в системе характеризуется состоянием. Состояние объекта – это множество запросов поступающих на сервис, а также характеристика текущей загруженности сервиса.

Объекты можно разделить на два подмножества: подмножество активных и подмножество пассивных объектов. Пассивный объект не может осуществлять доступ к другим объектам, тогда как активный может.

Обозначим:

Множество *SRV* - множество типов сервисов;

Множество *Asrv* – множество экземпляров активных сервисов;

Множество *Psrv* – множество экземпляров пассивных объектов РИС;

$$SRV = Asrv \cup Psrv$$

$$Asrv \cap Psrv = 0$$

Как уже было сказано, над любым объектом каждого типа определены операции доступа. Например, по чтению, записи, запуску на выполнение и т.п. Эти операции, как было отмечено выше, состоят из последовательности

элементарных операций, так называемых примитивов. Будем предполагать, что каждая операция доступа состоит из двух примитивов: открыть и закрыть. Например, в случае доступа по чтению – открыть такой-то объект на чтение, закрыть такой-то объект на чтение. Выполнить примитив «закрыть» можно только в том случае, если был ранее выполнен примитив «открыть». Обозначим a операцию доступа к объекту r, состоит из друх примитивов «открыть доступ по операции a», a ar - «закрыть доступ по операции a». Транзитивное замыкание описывает реально наблюдаемую ситуацию, когда один объект использует несколько других объектов через цепочку операций доступа. Например, клиент, отправляя запрос на комплексный сервис, который в свою очередь запускает на выполнение набор атомарных сервисов, которые в свою очередь могут запускать другие атомарные сервисы.

В том случае, если комплексный сервис, удовлетворяющий ограничениям, не будет найден при отборе составных (атомарных сервисов), для покрытия всей области решений по обеспечению заданного качества обслуживания необходима разработка метода повышения уровня QoS, в случае отсутствия сервисов с необходимым уровнем качества обслуживания.

Для решения поставленной задачи в работе предложен математический метод формирования распределенного комплексного сервиса на базе информации о имеющихся в сети атомарных сервисов

В основе предлагаемого метода лежит идея формирования комплексного сервиса R, состоящей из нескольких сервисов из множества FES с ПК ниже требуемых, одновременное использование которых позволит повысить доступность сервиса:

$$R_i \subset FES,\quad FES_l = \{R_1, R_2, \cdots, R_i, \cdots, R_n\},$$
$$R_i = \{S_w, S_{w+1}, \cdots, S_{w+y}\}, \qquad (2.14)$$

при этом возможна ситуация, когда $R_i \cap R_{i+1} \neq \varnothing$,

где i – номер комплексного сервиса в множестве FES, w – номер сервиса в выбранном подмножестве P_i, z – количество сервисов в подмножестве P_i, y – любое число от 1 до (z-w)

Распределенный комплексный сервис может быть представлен в виде обьединения множества сервисов *Pi*. При этом, QoS параметры атомарных сервисов Pi могут иметь значения ниже требуемых, однако, за счет их одновременного использования, суммарные показатели производительности распределенного сервиса возрастают.

Таким образом, в случае отсутствия сервиса с требуемыми показателями качества в программно-конфигурируемой сети применение метода формирования распределенного сервиса позволит сформировать сервис с заданными значениями параметров качества.

Значения учитываемых параметров QoS представлены множеством L. Информация о показателях QoS атомарных сервисов извлекается из модифицированого UDDI реестра [25].

Множеством K представлено значение параметра QoS, которое не удовлетворяет требованиям пользователя. Множество K принадлежит множеству L, $K \subset L$. Множество D также принадлежит множеству L, $D \subset L$ и включает в себя все параметры данного множества, за исключением параметра, принадлежащего множеству K, $D = L \setminus K$.

2.3. Модель обнаружения Slow HTTP атак

Исследования природы данного типа атак показали, что в процессе реализации поток заявок атакующего хоста можно считать простейшим. Об этом свидетельствует то, что исследуемый поток заявок обладает тремя свойствами, характерными простейшему потоку событий: стационарность, ординарность и отсутствие последствий.

Поток заявок атакующего хоста является стационарным, так как вероятность поступления и обработки заявок на любом промежутке времени зависит только от числа этих событий и от длительности промежутка, и не зависит от начала их отсчета.

Поток заявок атакующего хоста является ординарным, так как вероятностью поступления или обработки за элементарный промежуток времени более одной заявки можно пренебречь по сравнению с вероятностью поступления или обработки за этот промежуток не более одной заявки.

Поток заявок атакующего хоста обладает свойством отсутствия последействия, так как вероятность поступления или обработки числа заявок на любом промежутке времени не зависит от того, появлялись или не появлялись события в моменты времени, предшествующие началу рассматриваемого промежутка [19].

Параметры http-запросов (длинна, скорость приема данных, размер принимаемых данных, задержки между подтверждениями, методы запросов) так же являются одинаковыми и постоянными.

Данный факт позволяет описать атакуемый Web-сервер как систему массового обслуживания типа M/M/N, где N – максимальное количество одновременно обрабатываемых http-запросов (максимальное количество процессов (потоков), которые может запустить web-сервер). Например, для сервера Apache2 это параметр «MaxClients» в конфигурационном файле «http.conf».

Наличием буфера ожидания в web-сервере в данном случае можно пренебречь, так как он не влияет на факт начала атаки, а влияет лишь на ее продолжительность.

Следовательно, граф состояний атакуемого web-сервера выглядит, как представлено на рисунке 2.6.

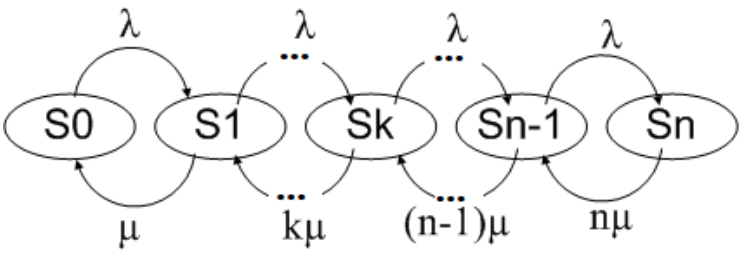


Рис. 2.6. Размеченный граф переходов состояний web-сервера

Состояниями web-сервера являются:

S_0 –нет обслуживаемых запросов;

S_1 - обслуживается 1 запрос;

S_k - обслуживается k запросов;

S_{n-1} - обслуживается $n-1$ запросов;

S_n - обслуживается n запросов, сервер находится в состоянии перегрузки.

Параметрами модели являются:

n- максимальное количество обслуживаемых http-запросов;

k- текущее количество http-запросов.

λ-интенсивность поступления http-запросов;

μ- интенсивность обслуживания http-запросов.

В соответствии с теорией массового обслуживания в один и тот же момент времени может произойти одно из двух событий, которые приводят к изменению состояния web-сервера:

– поступление http-запроса, приводящее к переходу в соседнее состояние с большим номером, причем если сервер находится в состоянии S_n, то его состояние не изменится, что соответствует отказу в обслуживании;

– завершение обслуживания http-запроса и переход в состояние с меньшим номером [20].

Приведенные рассуждения позволяют использовать формулы Эрланга для вычисления вероятности любого (k-го) состояния (p_k) web-сервера:

$$p_k = \frac{\frac{\alpha^k}{k!}}{\sum_{k=0}^{n}\frac{\alpha^k}{k!}}, \quad (2.14)$$

где:

$$\alpha = \frac{\lambda}{\mu}. \quad (2.15)$$

Выражение (3.2) позволяет связать интенсивность входного потока запросов (λ) в интенсивностью обслуженных запросов (μ) (закрытых соединений).

Особенностью Slow HTTP атак является не значительное увеличение интенсивности входного потока, а падение количества обслуженных http-запросов при стабильном входном потоке.

2.4 Анализ адекватности модели обнаружения Slow HTTP атак

Оценку точности и достоверности результатов имитационного моделирования предлагается производить с помощью оценки доверительного значения погрешности [10].

Практически это производится следующим образом.

1. Из всех полученных результатов отбрасываются наиболее удаленные от среднего как самые ненадежные.

2. Для оценки центра распределения вычисляются средние арифметические полученных значений:

$$\hat{P}_i^* = \frac{1}{n}\sum_{k=1}^{n} \hat{P}_{ik}^* , \qquad (2.16)$$

где $\hat{P}_i^*$ - оценка частоты появления ошибок i-й кратности;

n – число появлений анализируемых событий.

3. Доверительную вероятность (β) и допустимую величину относительной ошибки (ε) предложено установить: β = 0,8 и ε = 0,5%.

Дальнейшие вычисления производятся в предположении, что частоты появления различных событий (ошибки различной кратности, доведение или потеря пакета и т. п.) являются независимыми одинаково распределенными случайными величинами. Следовательно, согласно центральной предельной теореме, закон распределения суммы таких величин приближенно можно считать нормальным.

4. Вывод о адекватности модели делается при выполнении неравенства

$$P(|p - p^*| < \varepsilon) > \beta. \qquad (2.17)$$

5. На следующем шаге в зависимости от значения доверительной вероятности и допустимой величины ошибки вычисляются параметры имитационного эксперимента:

$$\varepsilon = \sigma_{\tilde{m}} arq\Phi^*(\frac{1+\beta}{2}), \text{ где} \qquad (2.18\_)$$

arqФ*() – функция, обратная функции Лапласа, значение данной функции для β=0,8 равно: tβ=1,282.

Данным требованиям соответствуют параметры эксперимента: k=100, m=10000, где k – число опытов, m – число исходов в опыте (переданных пакетов).

2.5 Оценка эффективности модели

При помощи инструмента «slow http test» было реализовано три типа низкоинтенсивных прикладных атак: Slowloris, Slow HTTP POST и Slow READ. Первые две атаки направлены на передачу HTTP-запроса с длительной задержкой между передачей частей запроса: в первом случае – медленная передача заголовка запроса, во втором случае – тела запроса. Третий тип атак медленно читает ответ сервера. Изначально Slow READ афиширует web-серверу большой размер TCP-окна приема, и, далее, меняет размер окна на очень маленький, что заставляет web-сервер постоянно находиться в режиме ожидания передачи данных. Все три типа атак, позволяют держать соединение с сервером открытым в течение длительного времени, позволяя им исчерпать все ресурсы web-сервера, не разрешая легальным пользователям получить услуги.

Для анализа эффективности разработанной модели проведен эксперимент в ходе которого генерировались различные сценарии slow-http атак на web-сервер. В ходе реализации атак оценивалась статистическая частота наступления различных состояний web-сервера.

В таблице 2.1 приведены основные параметры атаки на web-сервера «Apach 2» в момент отказа в обслуживании для вышеописанных типов Slow HTTP атак.

На рис. 2.7 показано распределение вероятностей состояний web-сервера в обычном режиме работы, а также в ходе реализации атаки Slow HTTP (Slow HEAD) при различных интенсивностях атаки.

Как видно из графика, представленного на рис. 2.7, в процессе реализации атаки распределение вероятностей состояний сервера сдвигается вправо.

Следовательно, система обнаружения Slow HTTP атак может выполнять расчет данного параметра в реальном масштабе времени и формировать предупреждающие уведомления.

Таблица 2.1

Основные параметры SlowHTTPатак

Тип атаки	**Slowloris**	**Slow POST**	**Slow READ**
Количество соединений	1000	410	1000
Тип запроса	GET	POST	-
Полученный диапазон окна	-	-	20-572
Количество одновременных запросов в одном соединении	-	-	1
Скорость чтения буфера приема	-	-	32байта/5 сек
Значение поля заголовка Content-Length (длина тела сообщения в байтах)	209	8192	-
Дополнительное поле данных	52	66	-
Интервал между последующими данными(сек)	5	10	-
Подключений в секунду	300	200	100
Тайм-аут для пробного соединения	5	5	5
Продолжительность тестирования(сек)	200	240	240
Использование прокси	нет	нет	нет

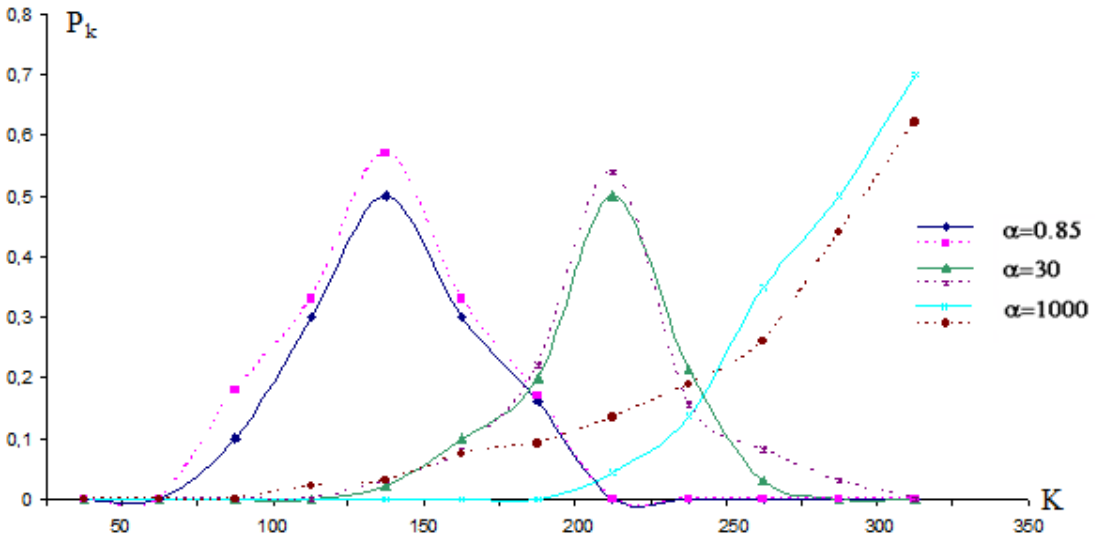


Рис. 2.7. Распределение вероятностей состояний web-сервера при различных значениях α

2.6. Выводы по разделу.

Анализ различных подходов к обнаружению DOS-атак на облачную инфраструктуру показал, что Марковские цепи являются напиболее подходящим аппаратом для описания web-сервера при реализации атаки.

Для разработки модели сервисов через аппарат Марковских цепей предложенно использовать подход описания поведения системы через набор состояний. Используя данный подход любой сервис в мультисервисной сети, в том числе и в облачной инфраструктуре может быть представлен через последовательность как своих локальных состояний, так и через последовательность состояний взаимодействия с другими компонентами инфраструктуры (другими сервисами).

В рамках решения задачи по обнаружению Slow-http атак выполнена разработка модели web-сервера на основе Марковской цепи.

В данной модели web-сервер представляется набором своих состояний соответствующих обслуживаемым запросам. Такое описание атакуемого объекта облачной инфраструктуры позволило получить распределение вероясностей состояний, в том числе и состояния «отказ в обслуживании» в зависимости от параметров slow-http атаки.

Проведение эксперимента по реализации slow-http атак в соответствии с основными сценариями подтвердило адекватность разработанной модели.

РАЗДЕЛ 3 РАЗРАБОТКА МОДЕЛИ ПРОГНОЗИРОВАНИЯ ВРЕМЕНИ ПЕРЕХОДА В ОТКАЗ В ОБСЛУЖИВАНИЯ

3.1 Анализ поведения сервиса и модель атаки

3.1.1 Анализ поведения сервиса

Проведение DOS-атаки на систему заключается в том, что ко входному потоку заданий, генерируемому обычными пользователями (далее будем называть этот поток регулярным), добавляется искусственный поток заданий требующих большого количества ресурсов. При отсутствии атак ресурсы системы, как правило, позволяют своевременно обслуживать все регулярные задания. Но при наличии большого количества искусственных заданий во входном потоке снижается эффективность обслуживания, поскольку помимо регулярных заданий приходится обслуживать и искусственные, поступающие с высокой интенсивностью и/или требующие значительного количества ресурсов. Вычислительная система не способна обрабатывать все поступающие задачи. В результате возрастает количество отказов в обслуживании, что и является целью атаки. Таким образом, возникает две задачи: зарегистрировать момент вторжения и изменить протокол обслуживания заданий так, чтобы продолжать обслуживать регулярные задачи в присутствии искусственного потока.

Описание динамики реализации slow-http атаки можно реализовать через описание траекторий изменения состояний атакуемого сервиса (объекта). Назовём траекторией t_r экземпляра объекта r некоторую непустую конечную последовательность состояний экземпляра объекта r, замкнутую слева, т.е. если $t_r = S_1, S_2, \dots, S_k \in S$ траектория, то $\forall i: i \leq k => S_1, S_2, \dots, S_k$ траектория.

Траекторию экземпляра активного объекта r можно представить как последовательность отрезков траекторий взаимодействующих с ним экземпляров объектов и собственных действий экземпляра объекта над экземплярами других объектов мультисервисной сети, так как любая операция открытия или закрытия доступа вызывает изменение состояния взаимодействующих экземпляров объектов. Траекторию экземпляра пассивного

объекта r можно представить, как последовательность отрезков траекторий взаимодействующих с ним экземпляров активных объектов.

Определим поведение экземпляра объекта как множество всех возможных его траекторий:

$$Bh(r) = \{t_r\}$$

Динамику поведения объекта можно определить, как динамику смены состояний среди множества доступных для объектов одного типа.

Траекторию сервиса в таком случае можно пределить как непустую конечную последовательность возможных состояний сервисов данного типа, замкнутую слева, в которой любые два соседних состояния отличны друг от друга.

Таким образом функционирование сервисов можно представить, как некоторую последовательность состояний, такую, что каждое последующее состояние отличается от предыдущего состоянием хотя бы одного из объектов.

Траекторию комплексного сервиса можно считать полной, если для любых двух соседних состояний S1 и S2 выполняются следующие условия:

1. $S_1 = \langle S_1, \cup S_i \rangle$ и $S_2 = \langle S_2, \cup S_i \rangle$

2. Для любого экземпляра объекта сервиса, такого что его состояния в соседних состояниях мультисервисной сети различны, существует взаимодействующий с ним экземпляр объекта сервиса в более раннем состоянии и его действие, вызвавшее смену состояний.

Эти условия означают, что если некоторая траектория полна, то она содержит все траектории экземпляров объектов, входящих нее. На множестве состояний отдельных экземпляров объектов, входящих в некоторую траекторию сервиса, можно ввести время, как отношение частичного порядка:

$$\forall S_{r,}^{i} S_{r,}^{j} \in T_{sys} : \exists r', a : S_{r,}^{i} \to S_{r,}^{j} => S_{r,}^{i} \prec S_{r,}^{j}$$

При этом будем говорить, что одно состояние есть причина, а другое - следствие, если в траектории первое состояние предшествует втором. Сервис находится в информационно безопасном состоянии, если все экземпляры объектов находятся в информационно безопасном состоянии. В соответствии со стандартом [8], информационно безопасное состояние – это множество таких

состояний, в которых отсутствуют следующие нарушения: нарушения конфиденциальности экземпляров объектов, нарушения целостности экземпляров объектов мультисервисной сети, нарушения доступности экземпляров объектов [25]. В терминах данной модели информационно безопасное состояние мультисервисной сети – это множество состояний сервисов, в которых все экземпляры сервисов, имеющие доступ к другим экземплярам сервисов, имеют права доступа, и загрузка каждого объекта меньше его ёмкости.

Пример опасного состояния: при исчерпании максимального числа открытых соединений можно говорить о наличии атаки, нарушающей доступность объекта (атаки типа DoS). Определим нарушение информационной безопасности как перевод сервиса из некоторого безопасного состояния в любое опасное.

Множество опасных состояний - это множество состояний, в которых хотя бы один экземпляр объекта находится в опасном состоянии. Переход любого экземпляра объекта в опасное состояние по определению означает переход всей мультисервисной сети либо комплексного сервиса в опасное состояние. Такое определение опасного состояния означает, что для его обнаружения нет необходимости собирать информацию о состоянии всех сервисов мультисервисной сети в каждый момент времени, а достаточно наблюдать только за изменениями состояний критичных, с точки зрения защиты информации, объектов.

Таким образом, атака – это траектория некоторого экземпляра объекта или набор участков траекторий некоторой группы экземпляров объектов, выводящая сервис из информационно безопасного состояния. Атака переводит систему из безопасного состояния в опасное, либо оставляет систему в ранее достигнутом опасном состоянии. Нормальное поведение объекта r определим, как множество траекторий всех экземпляров объектов данного типа *type(r)*, которые не выводят сервис из состояния информационной безопасности. Будем считать, что ни одна траектория, входящая в нормальное поведение объекта, не может содержать

опасных состояний, т.е. состояний, в которых ресурс перегружен или существует цепочка транзитивного доступа, нарушающая отношение прав доступа

3.1.2 Метод сбора информации о реализации атаки

Сбор информации о состоянии сервиса осуществляется путем анализа отклика данного сервиса на поступающие запросы от клиентов и других сервисов сети. По реакции на данные запросы контролируемые объекты посылают ответные запросы, включающие всю необходимую для оценки состояния информацию.

Рассмотрим процесс анализа состояния сервиса при взаимодействии его с другими сервисами.

При передаче тестового запроса через промежуточные сервисы сети вначале проверяется условие доступности этих сервисов. Пусть промежуточный сервис сети является свободным в данный момент времени с вероятностью $P_{своб}$. Если промежуточный сервис сети, через который должен проходить тестовый запрос, в данный момент времени занят, то через время $T_{та}$ тестовой запрос для k-го сервиса сети будет повторен. Вероятность занятости сервиса сети определяется как $P_{зан} = 1 – P_{своб}$.

Тестовый запрос может быть обработан на сервисе сети сети с искажением адреса контролирующего сервиса, искажением адреса вызываемого абонента (сервиса сети), принят правильно либо с ошибками. Если в тестовом запросе искажен адрес контролирующего сервиса либо адрес, то такой тестовой запрос можно считать потерянным и через время $T_{та}$ он будет повторен.

Если адреса обоих сервисов не искажены, то тестовой запрос можно считать принятым правильно с вероятностью $P_{пр}$, либо принятым с обнаруженными (с вероятностью $P_{оо}$) или необнаруженными (с вероятностью $P_{но}$) ошибками. Если ошибки в тестовом запросе были обнаружены, то его можно считать потерянным и через время $T_{та}$ он будет повторен. В случае же правильного приема тестового запроса или приема его с необнаруженными ошибками этот тестовой запрос будет отправлен дальше, на следующий промежуточный сервис.

Аналогичным образом происходит процесс передачи тестового запроса для k-го сервиса сети и на последующих промежуточных сервисах.

С контролируемого сервиса сети ответный запрос, содержащий информацию о состоянии этого сервиса, может быть отправлен в двух случаях: при правильном приеме k-ым узлом сети тестового запроса и при приеме его с необнаруженными ошибками.

Для оценки вероятностно-временных характеристик различных этапов реализации slow-http атаки, а так же оценки ее динамики выбран формальный аппарат вероятностно-временных графов. При этом этапы сбора информации о состоянии сервиса передачи управляющей информации на узлы системы формально могут быть представлены в виде таких графов, а их анализ и оценку будем проводить при помощи метода производящих функций [46].

3.2. Метод вероятностно-временных графов

Аппарат вероятностно-временных графов (ВВГ) подробно описан [38], при его использовании составляется ориентированный граф, вершины которого соответствуют состояниям моделируемой системы. Следовательно, вершинами такого графа могут быть соответствующие места Е-сети. Пары (P_{ij}, t_{ij}), описывающие дуги графа, определяют вероятность выбора дуги ij (P_{ij}) и время ее прохождения (t_{ij}). Для описания движения системы из начального состояния в конечное вводится функция дуги $f(P_{ij}, t_{ij})$. Вид этой функции должен быть таким, чтобы при нахождении произведений функций вероятности P_{ij} перемножались, а времена t_{ij} суммировались. Этим условиям удовлетворяет функция вида:

$$f_{ij}(P_{ij}, t_{ij}) = P_{ij} z^{t_{ij}}, \qquad (3.1)$$

где z – параметр.

Тогда функция последовательности смен состояний моделируемой системы может быть записана в виде:

$$f_{1,..k}(z) = \prod_{i=1}^{k} P_{i,i+1} z^{t_{i,i+1}} \qquad (3.2)$$

Производящая функция F(z), соответствующая графу, есть функция всех путей, соединяющих начальную и конечную вершины графа. Для упрощения

нахождения производящей функции выполняются эквивалентные преобразования графа. В ходе эквивалентных преобразований из исходного графа убираются промежуточные вершины, а функции дуг изменяются по заданным правилам.

Эквивалентные преобразования осуществляются до тех пор, когда можно будет написать производящую функцию, описывающую переход из начальной вершины графа в конечную, т.е. в графе должны остаться только вершины, соответствующие начальному состоянию моделируемой системы, и вершины, соответствующие конечным состояниям. Среднее время выполнения процесса ($T_{ср}$), описываемого таким графом, дисперсия ($D_{Tср}$) и вероятность достижения конечной вершины ($P_{квр}$) определяются по формулам:

$$T_{ср} = \frac{dF(z)}{dz}\Big|_{z=1};$$

$$D_{Tср} = \frac{d^2F(z)}{dz^2}\Big|_{z=1} + \frac{dF(z)}{dz}\Big|_{z=1} - \left(\frac{dF(z)}{dz}\Big|_{z=1}\right)^2; \qquad (3.3)$$

$$P_{квр} = F(z)\Big|_{z=1}.$$

3.2.1 Метод эквивалентных преобразований вероятностно-временных графов

Метод эквивалентных преобразований ВВГ разработан для обеспечения возможности уменьшения размерности графа и описания производящей функции перехода системы из начального состояния в конечное.

На рис. 3.1-3.5 представлены типовые структуры графов и приведены методы их упрощения.

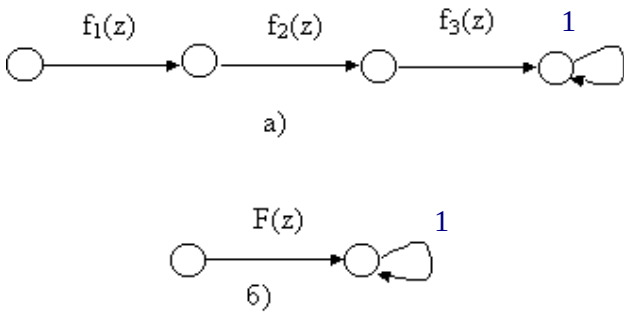


Рис. 3.1. Последовательное соединение вершин

Так, последовательное соединение вершин (рис. 3.1.а) с функциями дуг $f_1(z)=P_1z^{T1}$; $f_1(z)=P_2z^{T2}$; $f_1(z)=P_3z^{T3}$.

Эквивалентно одной дуге (рис. 3.1. б), производящая функция которой равна произведению $f(z)=f_1(z)f_2(z)f_3(z)=P_1z^{T1}\ P_2z^{T2}\ P_3z^{T3}$.

В общем случае для последовательно соединенных вершин

$$F(z)=\prod_i P_i z^{T_i}\ . \qquad (3.3)$$

Параллельное соединение дуг (рис. 2,а), с функциями дуг вида $f(z)= P_1z^{T1}$, эквивалентно одной дуге (рис. 2б) с результирующей функцией:

$F(z)=P_1z^{T1}+ P_2z^{T2}+ P_3z^{T3}$.

В общем случае при параллельном распределении дуг

$$F(z)=\sum_i P_i z^{T_i}\ . \qquad (3.4)$$

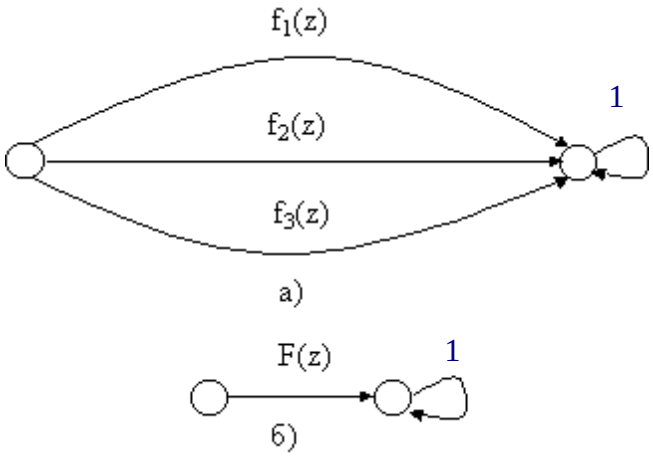


Рис. 3.2. Параллельное соединение дуг

Если же в графе имеется петля (рис. 3.3,а), то это характеризует бесконечно повторяющийся процесс, математически описываемый геометрической прогрессией.

Применительно к приведенному примеру знаменатель прогрессии равен функции $f_2(z)$, т. е. функции петли. Следовательно, при эквивалентных

преобразованиях граф, имеющий дугу-петлю, заменяется графом с дугой (рис. 3,б), описываемой функцией

$$F(z)=\frac{f_1(z)}{1-f_2(z)}. \qquad (3.5)$$

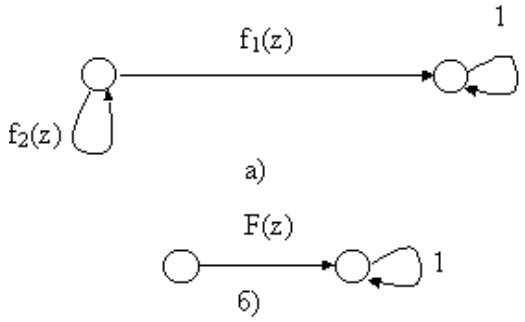


Рис. 3.3. Граф с дугой-петлей

Если в графе содержится петля в петле, как это показано на рис. 3.4,а, то сначала освобождаются от внешней петли. При этом дуги с функциями $f_3(z)$ и $f_4(z)$ заменяются одной дугой с производящей функцией вида

$$f'(z)=\frac{f_3(z)}{1-f_4(z)}.$$

После такого преобразования остается одна петля (рис. 4б), в которой последовательно соединены дуги с функциями $f_5(z)$ и $f'(z)$. На следующем этапе эти дуги заменяются одной (рис. 3.4,в) с производящей, равной произведению $f_5(z)$ и $f'(z)$: $f_Э(z)= f_5(z)f'(z)$.

В результате таких преобразований получается граф (рис. 3.4,в), аналогичный приведенному на рис. 3.3,а.

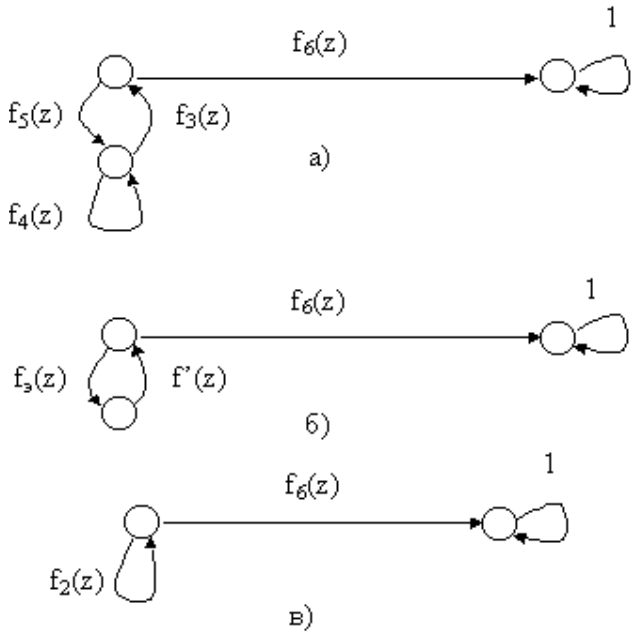
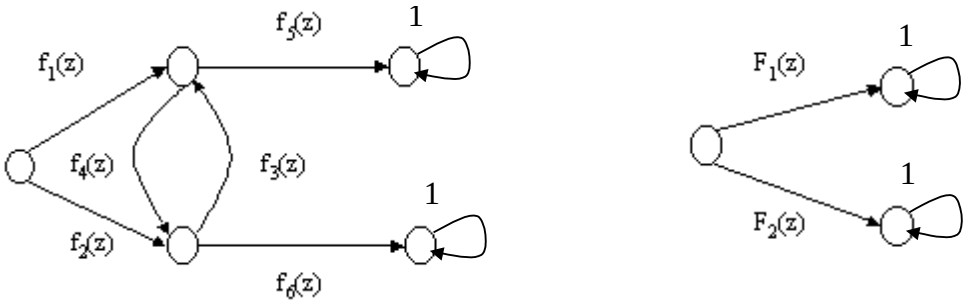


Рис. 3.4. Граф, имеющий вложенные петли

При наличии в графе промежуточных вершин, принадлежащих разным ветвям протекания моделируемого процесса, (например, как представлено на рис. 3.5,а), преобразование к виду (рис. 3.5,б) происходит в соответствии со следующими выражениями:

$$F_1(z) = \frac{f_5(z)}{1 - f_3(z) f_4(z)} (f_1(z) + f_2(z) f_3(z)); \qquad (3.6)$$

$$F_2(z) = \frac{f6(z)}{1 - f_3(z) f_4(z)} (f_2(z) + f_1(z) f_4(z));$$



а) б)

Рис. 3.5. Пример топологии ВВГ со связанными промежуточными вершинами

Таким образом, используя данный метод, обеспечивается возможность перехода к графам, содержащим только начальную и конечные вершины, и формирования результирующей производящей функции.

3.2.2. Метод анализа вероятностно-временных характеристик на «неприводимых» ВВГ.

Как выше было показано, при моделировании некоторых систем возможны такие топологии ВВГ (вероятностно-временных графов), которые методом эквивалентных преобразований нельзя привести к простейшим (содержащим только начальные и конечные вершины). Такие графы в дальнейшем будем называть «неприводимыми». Пример такого графа приведен на рис. 3.6.

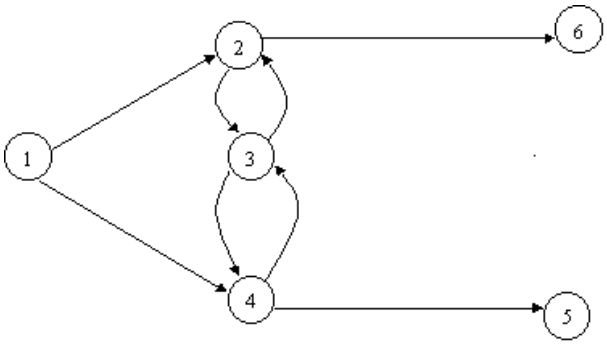


Рис. 3.6. Пример неприводимого графа

Как следствие, задача анализа вероятностно-временных характеристик (ВВХ) для таких графов представляется весьма сложной и трудоемкой. Следовательно, возникает необходимость в разработке метода, позволяющего с использованием вычислительных средств, реализовать анализ ВВХ систем, описываемых неприводимыми ВВГ.

Для решения поставленной задачи использованы элементы теории марковских процессов с дискретным состоянием и дискретным временем – марковских цепей.

Выбор данного аппарата обусловлен его схожестью с методом производящих функций, в частности, важным является то, что для обоих аппаратов понятие шага (такта) – переход системы из одного состояния в другое - является одинаковым.

При использовании аппарата марковских цепей распределение вероятностей между состояниями моделируемой системы на k-м такте находится из выражения

$$C_K = N \times P^K, \tag{3.7}$$

где P – матрица переходных вероятностей вида

$$P = \begin{pmatrix} p_{11} p_{12} & \cdots & p_{1n} \\ p_{12} p_{22} & \cdots & p_{2n} \\ p_{n1} p_{n2} & \cdots & p_{nn} \end{pmatrix}, \tag{3.8}$$

элементами данной матрицы являются вероятности p_{ij} перехода из вершины v_i в v_j, определенные ранее, при составлении и определении ВВГ. Свойства такой матрицы подробно описаны в [10];

$N = (p_1, p_2, \ldots, p_n)$ - вектор начальной разметки, в котором p_i – вероятность того, что система на нулевом шаге будет находиться в i-м состоянии: $p_i = P\{S_0 = \imath\}$. Для систем, описываемых ВВГ рассматриваемого типа, обычно вектор начальной разметки имеет вид: N=(0, …, 0, 1, 0,…,0), где p_i=1 означает то, что система на нулевом шаге будет находиться в начальном состоянии v_i, соответствующем вершине-истоку;

$C = (c_0, \ c_1, \ \cdots, c_i, \ \cdots, c_n)$ - вектор распределения вероятностей между состояниями моделируемой системы, элементами которого являются вероятности нахождения системы в состоянии v_i на k-м шаге ($p_i = P\{S_k = \imath\}$) (при моделировании СОД элементы вектора С, соответствующие конечным вершинам, могут интерпретироваться как вероятность правильного приема $P_{пр}$, вероятность ошибки $P_{ош}$).

Наличие вершины-истока и конечных вершин указывает на то, что системы, моделируемые графами такой топологии, через n шагов обязательно окажутся в одном из конечных состояний. Следовательно, с помощью аппарата цепей Маркова можно определить значение n.

Вычисление вектора C_K необходимо производить до тех пор, пока сумма вероятностей нахождения системы в конечных состояниях не будет больше

установленного исследователем порогового значения, номер шага на котором выполнится данное условие и принимается за конечный – n:

$$n = k \mid (P(S_k = i) + P(S_k = i) + .. P(S_k = l)) \geq g) \qquad (3.9)$$

где *i, j, l* – номера конечных вершин;

g – пороговое значение сумм вероятностей.

Таким образом, реализуется возможность определения с заданной точностью количества тактов, которое понадобится системе для перехода из начального состояния в одно из конечных.

Должен быть предусмотрен также режим работы, при котором исследователем задается не точность определения вероятностей конечных состояний, а количество тактов, после которого необходимо проанализировать распределение вероятностей между состояниями моделируемой системы.

Найденное число шагов n будет одинаково как для цепи Маркова, так и для ВВГ, но определение среднего времени достижения конечных вершин из начальной в данных математических аппаратах производится разными методами. При моделировании облачной инфраструктуры и ее компонентов определение временных характеристик с использованием аппарата цепей Маркова невозможно в силу того, что в данном аппарате время перехода системы между любыми двумя состояниями считается одинаковым или, для полумарковских цепей [11], распределенным по заданному закону распределения.

Определение среднего времени достижения конечных вершин с помощью метода производящих функций должно производиться следующим образом.

Исходя из того, что известно количество шагов n, которые система может пройти из начальной вершины, производится определение всех возможных траекторий движения системы: PU{(V_i,V_c) …}, где v_i – начальная вершина, v_c – любая вершина, достижимая из v_i за n шагов. Решение данной задачи основано на анализе матриц инцидентности A и смежности B. На первом шаге анализируется строка b_i матрицы смежности, соответствующая начальной вершине v_i. Целью такого анализа является определение всех вершин v_w, v_q, инцидентных начальной. На следующем шаге формируется k массивов,

соответствующих количеству инцидентных вершин, в которые заносятся номера дуг, связывающих каждую пару вершин (v_i, v_w). После этого анализируются строки матрицы смежности, соответствующие каждой исходящей вершине, определяются номера исходящих вершин для рассматриваемой. Затем на основании анализа матрицы инцидентности определяются номера связывающих их дуг, которые заносятся в уже сформированные массивы. В результате будут сформированы массивы PU, содержащие все возможные траектории движения системы за заданное количество шагов.

Затем из всего множества PU должны быть отобраны те пути (траектории), которые заканчиваются в конечных вершинах.

Любой из полученных путей E_{ic} можно представить последовательностью дуг, входящих в него:

$$E_{IC} = (e(v_i, v_k), e(v_k, v_l), e(v_l, v_c)),$$

следовательно, его можно представить как дугу, соединяющую начальную вершину V_K и конечную $V_{C.}$

3.3. Разработка метода прогнозирования времени реализации slow-http атаки

3.3.1. Метод оценки временных характеристик при реализации slow-http атаки

Принятие решения о возникновении атаки производится на этапах оценки ситуации и выбора средств и способов достижения цели.

При наличии полной и точной информации о состоянии сети ошибок при принятии решения не будет. Однако эти условия не выполняются. Поэтому возможны решения с ошибкой.

Как указывалось, ранее, сеть может находиться в следующих состояниях, характеризующих сложившуюся в ней ситуацию: норма; предупреждение; повреждение (затруднение управления); авария [77].

Сигналы формируются о состояниях контролируемых объектов данной системы:

«Норма» – параметры качества передачи и показатели режима и условий работы контролируемого объекта и его элементов находятся в допустимых пределах;

«Предупреждение» – параметры качества передачи находятся в установленных пределах, а показатели качества работы контролируемого объекта говорят о повышенной возможности его отказа;

«Повреждение» («Затруднение управления») – параметры качества передачи вышли за установленные границы в результате нарушения режима или условий работы контролируемого объекта и его элементов, но наличие повреждений в нем позволяет частично использовать объект в сети;

«Авария» – определенные для контроля параметры качества передачи вышли за установленные границы в результате нарушения режима или условий работы контролируемого объекта, наличие повреждений на нем приводит к невозможности его использования в сети.

При принятии решения на управление сетью, т.е. при оценке ситуации и выборе средств и способов достижения цели в указанных ситуациях, возможно одновременное выдвижение нескольких гипотез (альтернатив) с последующей отбраковкой или попарное выдвижение гипотез (альтернатив). Разработаем метод оценки временных характеристик при принятии решения для указанных вариантов и сравним эти варианты с целью выработки практических рекомендаций по их применению. С целью упрощения процесса анализа в дальнейшем как при оценке ситуации, так и при выборе средств и способов достижения цели будем полагать возможное выдвижение только трех таких гипотез. Принятое ограничение не сказывается на общности полученных выводов.

Представим сложившуюся на сети ситуацию в виде графа (рис.3.6). Обозначим:

вершина 0 – начальное состояние;

вершина 1 – выдвижение одной истинной гипотезы (альтернативы);

вершина 1' – выдвижение одной ложной гипотезы (альтернативы);

вершина 2 – выдвижение двух гипотез, одна из которых истинная;

вершина 2' – обе выдвигаемые гипотезы (альтернативы) ложные;

вершина 3 – выдвижение трех гипотез, одна из которых истинная.

После дальнейшей отбраковки из вершины 3 по дугам h32 и h32, можно перейти соответственно в вершины 2 и 2' , а затем по дугам h21, h21,; h2,0, h2,1, соответственно в вершины 1, 1’ или 0 (вершина 0 – исходное состояние). В состоянии, характеризуемом вершинами 1 и 1’ принимается окончательное решение.

В результате из вершины 1' сеть может перейти в вершину “Ошибка” по дуге h1,4, по дуге h1,0 – в вершину 0. Из вершины 1 сеть может перейти в вершину “Правильное решение” по дуге h14 и по дуге h10 в вершину 0.

Так как информация о ситуации в сети носит не полностью определенный характер, т.е. является нечеткой, то нельзя указать конкретные значения вероятностей перехода из одной вершины графа в другую. Поэтому воспользуемся значениями функций принадлежности этих вероятностей перехода, определяющими интервал значений той или иной вероятности [37].

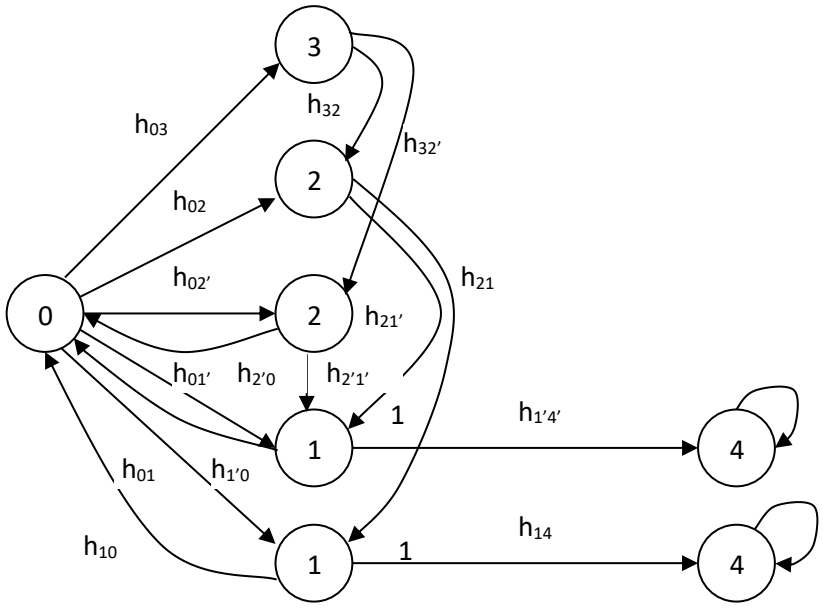


Рис.3.6. Граф, описывающий процесс принятия решения

Обозначим $\mu(P_1)$ и $\mu(P_{1'})$ – соответственно функции принадлежностей вероятности выдвижения за время t1 одной истинной и одной ложной гипотезы.

$\mu(P_2)$ и $\mu(P_{2'})$ – функции принадлежности вероятности выдвижения двух альтернатив соответственно, одна из которых истинная, и обоих ложных.

$\mu(P_3)$ – функция принадлежности вероятности выдвижения трех гипотез, одна из которых истинная.

Функции принадлежности и временные характеристики определяют ошибки управления и должны быть отображены в базе знаний.

Применим аппарат функций принадлежности для определения переходов из одной вершины графа (начальной) в другие.

Тогда:

$$h_{03}(z) = \mu(P_3)\cdot z^{t_1};\quad h_{02}(z) = \mu(P_2)\cdot z^{t_1};\ h_{02'}(z) = \mu(P_{2'})\cdot z^{t_1};$$

$$h_{01}(z) = \mu(P_1)\cdot z^{t_1};\quad h_{01'}(z) = \mu(P_{1'})\cdot z^{t_1}. \qquad (3.7)$$

Далее обозначим:

$$h_{32}(z) = \mu(P_{32})\cdot z^{t_2};\ h_{32'}(z) = \mu(P_{32'})\cdot z^{t_2};\ h_{21}(z) = \mu(P_{21})\cdot z^{t_2};$$

$$h_{21'}(z) = \mu(P_{21'})\cdot z^{t_2};\ h_{2'1'}(z) = \mu(P_{2'1'})\cdot z^{t_2};\quad h_{2'0}(z) = (1-\mu(P_{2'1'}))\cdot z^{t_3}$$

$$h_{1'0}(z) = (1-\mu(P_{1'4'}))\cdot z^{t_3};\ h_{10}(z) = (1-\mu(P_{14}))\cdot z^{t_3}. \qquad (3.8)$$

В формулах (3.8) приняты следующие обозначения:

$\mu(P_{32})$ – функция принадлежности вероятности перехода из вершины 3 в вершину 2 графа;

$\mu(P_{32'})$ – функция принадлежности вероятности перехода из вершины 3 в вершину 2' графа;

$\mu(P_{21})$ – функция принадлежности вероятности перехода из вершины 2 в вершину 1 графа;

$\mu(P_{21'})$ – функция принадлежности вероятности перехода из вершины 2 в вершину 1' графа;

$\mu(P_{2'1'})$ – функция принадлежности вероятности перехода из вершины 2' в вершину 1'.

И, наконец:

$$h_{14}(z) = \mu(P_{14}) \cdot z^{t_3}; h_{1'4'}(z) = \mu(P_{1'4'}) \cdot z^{t_3}. \quad (3.9)$$

При наличии четкой информации эти выражения примут вид:

$$h_{03}(z) = P_3 \cdot z^{t_1}; \quad h_{02}(z) = P_2 \cdot z^{t_1}; \quad h_{02'}(z) = P_{2'} \cdot z^{t_1};$$

$$h_{01}(z) = P_1 \cdot z^{t_1}; \quad h_{01'}(z) = P_{1'} \cdot z^{t_1}. \quad (3.10)$$

$$h_{32}(z) = P_{32} \cdot z^{t_2}; h_{32'}(z) = P_{32'} \cdot z^{t_2}; \quad h_{21}(z) = P_{21} \cdot z^{t_2};$$

$$h_{21'}(z) = P_{21'} \cdot z^{t_2}; h_{2'1'}(z) = P_{2'1'} \cdot z^{t_2}; \quad h_{2'0}(z) = (1 - P_{2'1'}) \cdot z^{t_3};$$

$$h_{1'0}(z) = (1 - P_{1'4'}) \cdot z^{t_3}; h_{10}(z) = (1 - P_{14}) \cdot z^{t_3}. \quad (3.11)$$

$$h_{14}(z) = P_{14} \cdot z^{t_3}; \quad h_{1'4'}(z) = P_{1'4'} \cdot z^{t_3}. \quad (3.12)$$

При наличии конкретной информации о состоянии сети обычно функции дуг h_{03}, h_{02}, $h_{02'}$, h_{01} и $h_{01'}$ известны. Требуется выбрать вариант, обеспечивающий минимальное время решения при заданной вероятности ошибки, или наоборот.

Пусть время, затрачиваемое на выдвижение альтернатив при принятии решения о состоянии сети, не превышает t_1 единиц. Это можно объяснить тем, что, проанализировав состояние сервиса по первичным признакам, система с большой долей вероятности не может принять решение о том, что сеть находится в состоянии “Авария”, если первичные признаки не дают об этом хотя бы малейшей информации. И наоборот, если первичные признаки состояния сервиса сигнализируют о каких-либо признаках наличия slow-http атаки, нельзя сделать вывод, что сервис находится в состоянии “Норма”.

Таким образом, остается наиболее вероятным случай выдвижения трех альтернатив (при условии, что сеть может находиться в одном из четырех вышеуказанных состояний), одна из которых в итоге окажется истинной (“Норма”, “Предупреждение”, “Затруднение управления” или “Предупреждение”, “Затруднение управления”, “Авария”).

Время анализа выдвинутых альтернатив (отбраковка ложных либо, с учетом стрессовой ситуации и возможной недостаточной квалификации, истинной гипотезы) или требуемое время на получение новой информации предположим не превышающим $2t_1$ единиц, т.е. соответствующим двум временам выдвижения самих альтернатив.

И, наконец, время на принятие окончательного решения, исходя из выдвигаемых требований, или требуемое время на получение дополнительной информации не должно превышать $3t_1$ единиц, т.е. пусть $t_2 = 2 \cdot t_1,\ t_3 = 3 \cdot t_1.$

Вероятность выдвижения альтернатив (одной из пяти: P_1, $P_{1'}$, P_2, $P_{2'}$ или P_3) предположим, как изменяющуюся в пределах от 0,1 до 1. При этом остальные случаи выдвижения альтернатив пусть будут равновероятными.

Согласно условию нормировки: $P_1+P_{1'}+P_2+P_{2'}+P_3=1$.

Тогда, если $P_1=0{,}1 \ldots 1$, то $P_{1'}= P_2 = P_{2'} = P_3 = (1–P_1)/4$.

Вероятность отбраковки ложной гипотезы при выдвижении двух альтернатив также возьмем равной $P_{21}=0{,}9$ (эта же вероятность соответствует случаю отбраковки ложной гипотезы при выдвижении трех альтернатив, если при этом одна ложная гипотеза уже была отброшена). Тогда: $P_{21'} = 1–P_{21} = 0{,}1$.

Вероятность возвращения в исходное состояние в случаях выдвижения двух или одной ложных альтернатив соответственно возьмем равной 0,9, т.е. $P_{2'0} = P_{1'0} = 0{,}9$. Тогда, согласно условию нормировки, получим: $P_{2'1'} = 1– P_{2'0} = =0{,}1$ и $P_{1'4'} = 1– P_{1'0} = 0{,}1$.

Вероятность возвращения в исходное состояние в случае одной истинной альтернативы примем равной 0,1. Тогда: $P_{14} = 1– P_{10} = 0{,}9$.

На рис.3.7 показаны графические зависимости относительного среднего времени принятия решения о реализации slow-http атаки.

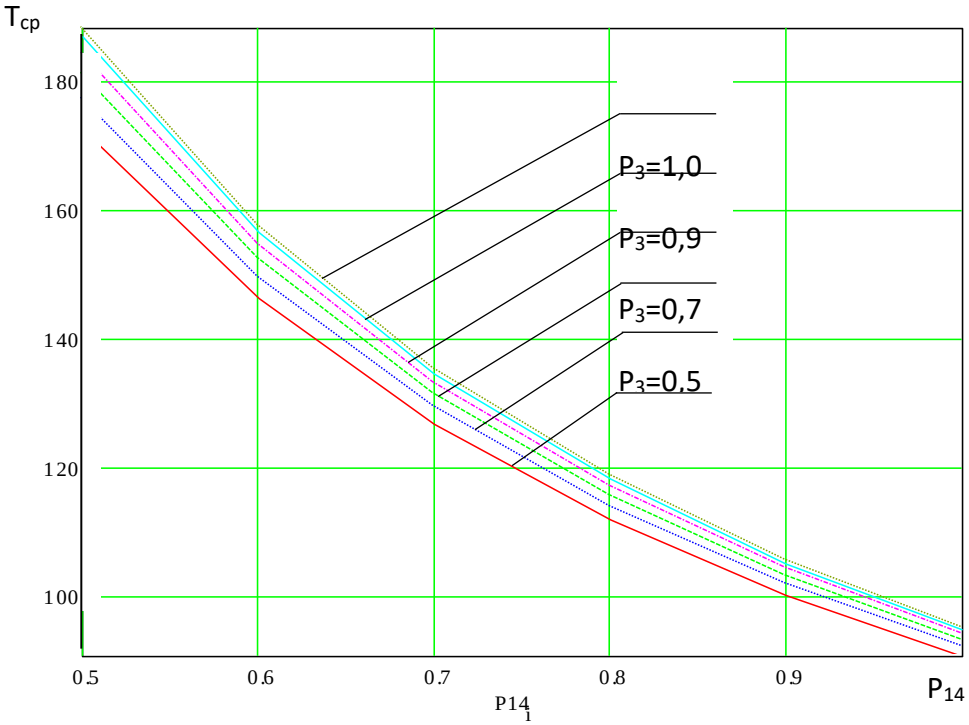


Рис.3.7. График зависимости $T_{ср}=f(P_{14})$ при разных значениях вероятности выдвижения трех гипотез, одна из которых истинная

3.3.2. Модель анализа динамики slow-http атаки.

В соответствии с методом производящих функций система представляется набором состояний и характеристиками переходов между ними, записываемыми в виде (3.1).

Отображая полученную ранее (п. 2.2) модель состояний web-сервера, представленную в виде марковской цепи на модель с использованием метода производящих функций, получен вероятностно-временной граф, представленный на рисунке 3.8:

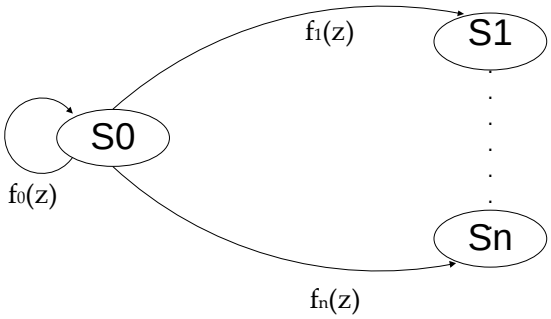


Рис. 3.8. Вероятностно-временной граф переходов состояний web-сервера

В соответствии с методами преобразования вероятностно-временных графов, он приводится к виду, представленному на рис. 3.9.

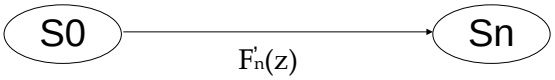


Рис. 3.9. Преобразованный вероятностно-временной граф переходов состояний web-сервера

Производящей функцией данного графа, будет функция вида:

$$F_n'(z) = (\sum_{i=1}^{n} P_i z^{t_i}) * (1 - P_0 z^{t_0})^{-1}. \quad (3.10)$$

Используя данное выражение, можно определить время перехода атакуемого сервера в состояние перегрузки:

$$T_{\text{ср}n} = \frac{dF_n'(z)}{dz}\Big|_{z=1} = \frac{(\sum_{i=1}^{n} P_i t_i)*(P_0-1)+\sum_{i=1}^{n} P_i * P_0 t_0}{(1-P_0)^2}, \quad (3.11)$$

На рис. 3.10 представлена зависимость времени перехода web-сервера в состояние перегрузки от отношения интенсивности входного и выходного потоков при различных объемах параллельно обслуживаемых запросах.

Анализ характера зависимости показал, что она является нелинейной, в большей степени определяется соотношением входного потока и потока обслуживания и практически не зависит от параметров сервера, определяющих количество параллельно обслуживаемых запросов.

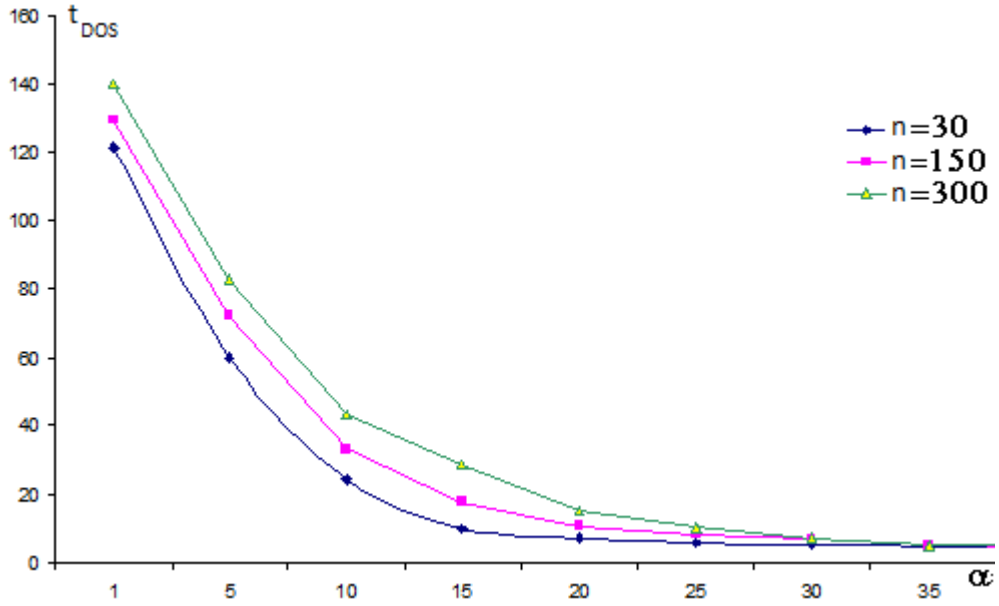


Рис. 3.10. Зависимость времени перехода web-сервера в состояние перегрузки, от отношения интенсивности входного и выходного потоков

3.4. Выводы по разделу

Анализ моделей обнаружения как обычных DOS-атак, так и Slow-HTTP показал, что они позволяют обнаружить лишь факт реализации атаки. В тоже время для повышения точности работы системы обнаружения и предотвращения атак небходимо так же знать и временные параметры. Например время перехода защищаемого объекта в состояние «отказ в обслуживании».

Для получения временных характеристик реализании slow-HTTP атаки предложенно использовать аппарат вероятностно-временных графов. Выбор данного математического аппарата в первую очередь обусловлен тем, что в качестве входных параметров может выступать модель состояний web-сервера на основе Марковской цепи.

Разработка модели прогнозирования перехода web-сервера в состояние «отказ в обслуживании» позволит повысить точность обнаружения и классификации атаки, а следовательно и выработать адекватные защитные действия системы обнаружения вторжений.

РАЗДЕЛ 4 МЕТОД ОБНАРУЖЕНИЯ И КЛАССИФИКАЦИИ SLOW-HTTP АТАК

4.1. Модель безопасности процесса обработки данных в облачных системах.

На сегодняшний день общим недостатком существующих систем обнаружения и предотвращения сетевых атак является отсутствие системного подхода к анализу поведения защищаемой системы в внешней среды. Другим недостатком является то, что они в основном ориентированы за защиту традиционной физической инфраструктуры и не учитывают особенности облачных систем.

Облачная инфраструктура подразумевает под собой особую клиент-серверную технологию, которая включает использование пользователем ресурсов не конкретного устройства, а группы устройств взаимодействующих между собой. В данной структуре пользователь может гибко управлять объемом и составом потребляемых ресурсов. Данный подход дает пользователю значительную свободу и гибкость, с другой стороны возлагает повышенные требования к системам сетевой защиты, в частности к системам обнаружения и предотвращения сетевых атак.

Таким образом перед разработкой метода обнаружения и предотвращения вторжений необходимо детально описать процессы, протекающие в облачной инфраструктуре при обработке данных.

Для определения модели безопасности рабочих процессов в облачной среде предлагается взять за основу модель безопасности Белла-ЛаПадула [45]. В соответствии с данной моделью система представляется в виде субъектов{S}, объектов {O} и прав доступа {R}. Уровни безопасности субъектов и объектов задаются с помощью функции безопасности.

Функция безопасности F назначает каждой паре элементов S и O некоторый уровень из L, разбивая множество сущностей системы на классы, в

пределах которых их свойства с точки зрения модели безопасности являются эквивалентными:

$$F: S \cup O \to L$$

Относительно облачной инфраструктуры данную модель предлагается расширить за счет добавления новых условий и компонентов отражающих особенности данной области.

Процесс обработки данных представляется в виде ориентированного графа вершинам которого соответствуют отдельные сервисы. Дуги графа соответствуют функциональным связям между сервисами. Разработка данной модели позволяет выполнить анализ потенциальных уязвимостей облачной инфраструктуры с точки зрения сетевых атак. Выделить периферийные сервисы, которые взаимодействуют с конечными пользователями и следовательно уязвимы для DOS-атак, корневые сервисы, функционирование которых влияет на всю инфраструктуру в целом.

Так на рис. 1.1 приведена архитектура облачной инфраструктуры OpenStack, соответствующий ей граф обработки представлен на рис. 4.1.

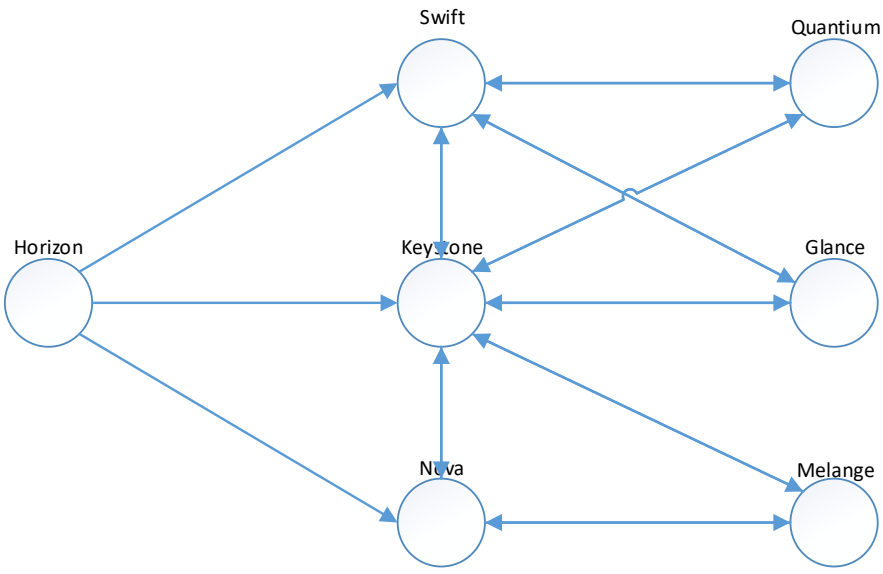


Рис. 4.1. Граф обработки в OpenStack

Используя полученный граф обработки можно определить все варианты обработки данных в рамках среды OpenStack (рис. 4.2).

Анализ данного графа позволил определить, что периферийными сервисами является Horizon, через данный сервис осуществляется все взаимодействие системы и внешней среды. Следовательно, данный сервис должен быть в первую очередь защищен от сетевых атак, включая Slow-HTTP.

В тоже время сервисы Keystone и Nova являются корневыми сервисами, которые определяют весь процесс взаимодействия компонентов облачной инфраструктуры между собой. Следовательно, вопросам защиты данных сервисов так же должно быть уделено соответствующее внимание.

4.2. Метод обнаружения и классификации Slow-HTTP атак на облачную структуру OpenStack

4.2.1. Анализ архитектуры OpenStack

Nova (Compute) – контроллер, управляющий работой виртуальных машин. Nova сосредоточена на таких функциях, как обработка запросов на создание виртуальных машин, соединение их с внешним миром, контроль за работоспособностью и распределением нагрузки на физические машины и каналы связи, реакция на сбои и т.д. В основе Nova лежит код системы NASA Nebula, язык программирования Python и протокол обмена сообщениями AMQP.

В системе существует восемь обособленных компонентов:

- Контроллер Облака (Cloud Controller) следит за состоянием системы и является связующим звеном всех остальных компонентов.
- Сервер API (API Server) реализует web-интерфейс, позволяющий управлять контроллером облака.
- Контроллер вычислений (Compute Controller) отвечает за запуск виртуальных машин и их связь со всей остальной инфраструктурой.
- Хранилище (Object Store) предоставляет сервис хранения данных, совместимый с Amazon S3.
- Менеджер аутентификации (Auth Manager) предоставляет сервисы аутентификации и авторизации.
- Контроллер томов (Volume Controller) дает возможностьподключать виртуальные устройства хранения к виртуальным машинам.

- Сетевой контроллер (Network Controller) создает виртуальные сети, позволяя виртуальным машинам взаимодействовать друг с другом и с внешней сетью.
- Планировщик (Scheduler) ответственен за выбор подходящего контроллера вычислений для запуска новой виртуальной машины.

Перечисленные выше компоненты связаны между собой.

«Управляющий центр» облака. В «управляющий центр» облака, работающий на выделенной машине, входит сервер API, контроллер облака и менеджер аутентификации. Администратор использует утилиту nova-manage для управления характеристиками всей инфраструктуры и доступом к ней пользователей. Клиенты, которые хотят использовать облачный сервис, подключаются к серверу API с помощью клиентских утилит Amazon EC2 или их свободного варианта под названием euca2ool из проекта Eucalyptus.

Управление хранилищем данных (ObjectStore), за которое отвечает еще одна выделенная машина. В состав Nova включена только начальная реализация S3-совместимого хранилища данных, использующаяся только для отладки. В реальных проектах на ее месте должен быть установлен Swift, развивающийся отдельно от Nova.

Контроллер томов (еще одна выделенная машина) позволяет подключать к виртуальным машинам своего рода внешние накопители данных (виртуальные флэшбрелки). Его присутствие в инфраструктуре необязательно.

За распределение IP-адресов между виртуальными машинами отвечают несколько машин, которые выполняют работу сетевых контролеров. Они же могут выступать в роли шлюза, отделяющего виртуальные машины от остальных сегментов сети. Как правило, на один сегмент внутренней виртуальной сети приходится один сетевой контроллер.

Одна из машин выполняет функции планировщика, который следит за контроллерами вычислений. Когда поступает новый запрос на создание виртуальной машины, планировщик выбирает для этой цели наиболее подходящего кандидата (решение о пригодности может быть принято на основе

загруженности контроллеров вычислений, количества виртуальных машин, выполняемых на них, и других факторов).

Основной костяк облачной инфраструктуры OpenStack – это контроллеры вычислений. Как правило, их количество превосходит количество всех остальных машин сети. В функции контроллеров входит прием запросов на создание новой виртуальной машины, ее запуск, слежение за состоянием виртуальных машин, перезапуск и так далее. Количество контроллеров вычислений в инфраструктуре напрямую определяет количество клиентов, которых может обслуживать сервис.

Swift (OpenStack Object Storage) — это распределенное хранилище данных, которое характеризуется отказоустойчивостью и высокой надежностью.

Четыре основных компонента Swift:

- Proxy Server (прокси-сервер): объединяет все компоненты системы вместе.
- Object Server (объектный сервер): ответственен за хранение данных.
- Container Server (контейнерный сервер): в его функции входит отдача списка объектов.
- Account Server (сервер аккаунтинга): отдает листинги контейнеров для конкретного аккаунта.

Типичная Swift-инфраструктура представляет собой кластер, одна из машин которого выполняет функции прокси-сервера, несколько машин работают в качестве контейнерных серверов и серверов аккаунтинга, а все остальные (сотни и тысячи машин) представляют собой контейнерные серверы.

Glance – модуль отбора, регистрации и поиска образов виртуальных компьютеров «machine images» (VMI). В рамках Glance используется RESTful API, что позволяет делать запрос метаданных VMI и выполнять поиск фактического образа (VMI).

Связь с Glance осуществляется посредством интерфейса HTTP по типу REST.

Кроме того, Glance использует клиентский класс, который обеспечивает простую и быструю работу. Также для взаимодействия с Glance в версии Cactus предлагается набор инструментов, обеспечивающих работу по командам.

Horizon – модуль, через который можно осуществлять взаимодействие с различными сервисами OpenStack, например, запустить инстанс (или виртуальную машину, VM), получить доступ к подчиненному хранилищу файлов, попробовать задать разные сетевые настройки (IP-адреса, доступы и т.д.). Horizon, главным образом, работает как инструмент пользовательского интерфейса для общения с более высокопроизводительными сервисами, такими как Nova, Swift и др. Аналогом данного компонента в VMware vCloud является vCloud Director User Interface Console. Чтобы начать работу с Horizon, необходимы компонент для управления ID, осуществляющий аутентификацию (в OpenStack это KeyStone), и система управления образами.

KeyStone – это компонент, который предоставляет услуги идентификации, как ILM в Microsoft Networking. Однако, в отличие от службы Active Directory (которая работает на основе маркеров), он также обеспечивает централизованный механизм аутентификации для облака OpenStack в формате имя пользователя /пароль. Без этого пользователь даже не сможет выполнить вход в панель управления Horizon и начать использовать облачные сервисы в OpenStack. В свою очередь, данный сервис осуществляет подключение к Active Directory или OpenLDAP или даже Amazon AWS внутренней корпоративной сети. KeyStone использует OpenStack Identity Service API, который реализован через web-интерфейс сервиса RESTful. Он работает по протоколу SSL поверх HTTP (HTTPS) через TCP-порт 443.

4.2.2. Разработка метода обнаружения и классификации Slow-HTTP атак

Анализ архитектуры и потоков обработки в облачной инфраструктуре OpenStack позволил выделить наиболее уязвимые компоненты подверженные атакам на отказ в обслуживании, в том числе и Slow-HTTP атакам.

На рис. 4.2 приведена диаграмма обслуживания запроса пользователя на обращение к какому-либо сервису.

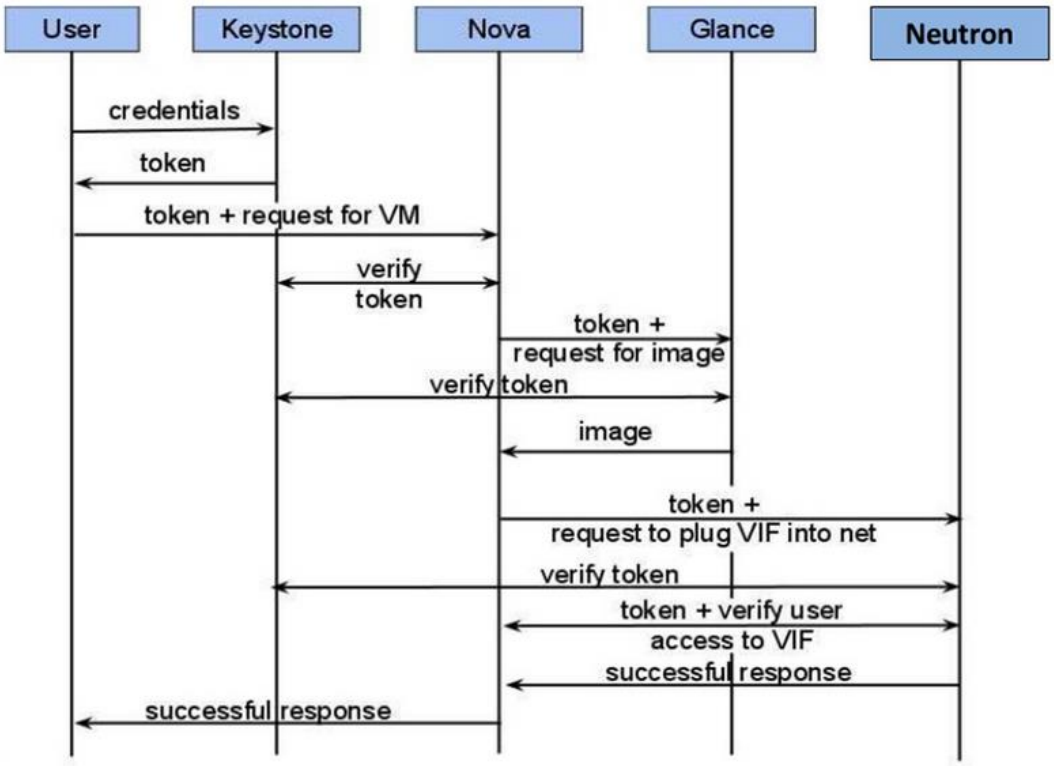


Рис. 4.2. Диаграмма обслуживания запроса в инфраструктуре OpenStack

Анализ диаграммы показывает, что все запросы, поступающие на облачную инфраструктуру, проходят через модуль Keystone. Следовательно, данный модуль и будет наиболее подвержен опасности выхода в состояние «отказ в обслуживании».

В качестве основы метода обнаружения slow-HTTP атак на облачную инфраструктуру типа OpenStack предлагается использовать модель обнаружения атаки на базе состояний веб-сервера, разработанную п. 2.2. В настоящее время так же значительную популярность получили фрактальные модели, однако в данном случае их использование не представляется возможным в силу малого интервала наблюдения.

Реализацию отдельно взятой атаки на компоненты облака можно представить как СМО с очередью длиной m. Длина очереди определяется размером входного буфера ожидания каждого отдельного компонента.

Рассматривая отдельные компоненты системы можно предположить, что они функционируют в стационарном режиме. Поток заявок на узел принимается простейшим с интенсивностью λ.

Узел обслуживает заявки с интенсивностью μ. После обслуживания заявки она полностью покидает систему. Если прибывшая заявка обнаруживает, сто входная очередь ожидания заполнена, то заявка теряется. Следовательно, найдя вероятность и время потери заявки – найдем вероятность и время реализации атаки.

На облако поступает вектор входных заявок с интенсивностью $\vec{\lambda}$. Которые обслуживаются с интенсивностью $\vec{\mu}$. Переходы между узлами осуществляются с вероятностями $p_{i,j}$ (рис. 4.3)

Данную систему можно описать дискретной цепью Маркова, соответствующей движению заявки по компонентам облака (п. 2.2).

Каждая вершина цепи Маркова характеризуется количеством заявок находящихся на обслуживании в этом узле ($d \in 1, m_i + 1$).

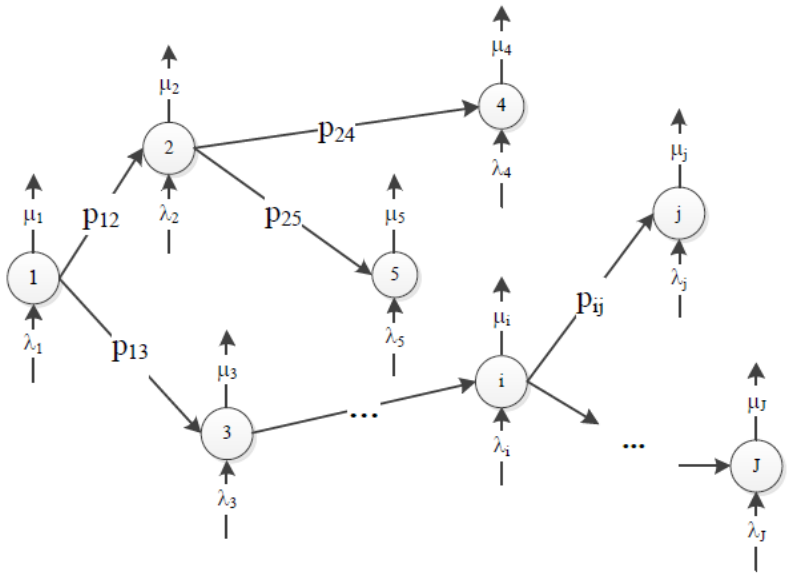


Рис. 4.3. Граф реализации сетевой атаки на элементы облачной инфраструктуры

Начальное распределение данной цепи, можно рассчитать на основе выражения:

$$p_i = \frac{\lambda_i \frac{p_i^{d-1}}{\mu_i^{d-1}}\left(1-\frac{p_i}{\mu_i}\right)}{\sum_{j=1}^{J} \lambda_{j^*}\left(1-\frac{p_i^{m_i}}{\mu_i^{m_i}}\right)} \quad (4.1)$$

Матрица вероятностей переходов описанной цепи определяется следующим образом:

$$p(i,j) = p_{i.j}\frac{\frac{p_j^{d_j-1}}{\mu_j^{d_j-1}}(1-\frac{p_j}{\mu_j})}{1-\frac{p_j^{m_i-1}}{\mu_j^{m_i-1}}} \quad (4.2)$$

Определение времени перехода в состояние «отказ в обслуживание» выполняется на основе модели, разработанной в п. 3.2.

Для определения времени перехода в состояние «отказ в обслуживание» цепь Маркова необходимо привести к вероятностно-временному графу представленному на рис. 4.4.

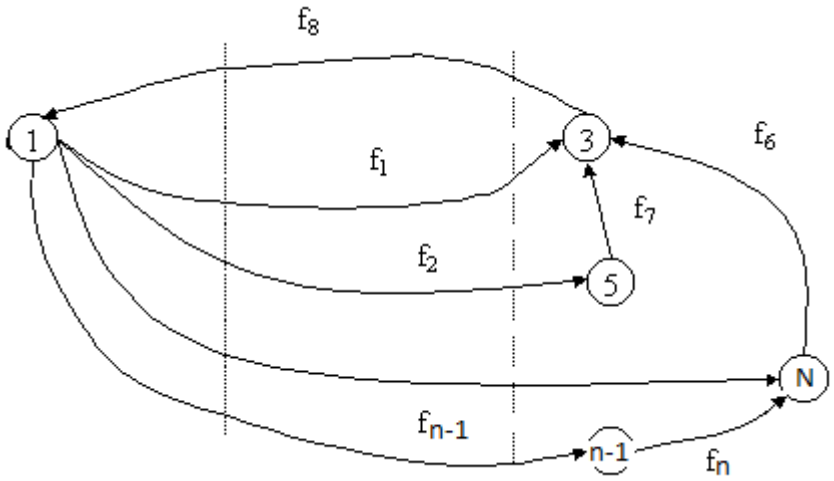


Рис. 4.4. Вероятностно-временной граф переходов модуля Keystone

Состояния графа соответствуют состояниям цепи Маркова, переходы между состояниями описываются выражением вида:

$$F_n'(z) = (\sum_{i=1}^{6} P_i z^{t_i}) * (1 - P_0 z^{t_0})^{-1} \quad (4.3)$$

$$T_{dos} = \frac{dF_n'(z)}{dz}|_{z=1} = \frac{(\sum_{i=1}^{6} P_i t_i) * -(1-P_0) + \sum_{i=1}^{6} P_i * P_0 t_0}{(1-P_0)^2}, \text{ где} \quad (4.4)$$

$$t_i = \frac{1}{\lambda} k_i$$

4.3. Система обнаружения и предотвращения Slow-HTTP атак

На основе модели, описанной в пункте 4.2, разработана система обнаружения Slow HTTP атак, функциональная схема работы которой представлена ниже на рисунке 4.5.

Рис. 4.5. Функциональная схема системы обнаружения Slow HTTP атак

Данная система состоит из четырех взаимосвязанных модулей:

- модуль сбора трафика;
- модуль формирования сетевой статистики;
- модуль расчета загрузки и перегрузки web-сервера;
- модуль принятия решения.

Процесс работы каждого модуля задается определенными алгоритмами, описание которых представлено ниже:

1) Алгоритм сбора сетевого трафика (рис 4.6).

Первый этап обнаружения Slow HTTP атак заключается в накапливании трафика модулем сбора сетевой статистики. Данный модуль собирает пакеты в течении некоторого промежутка времени, выделяет параметры, необходимые для дальнейших расчетов СОА и записывает их в массив структур: IP-адрес источника, IP-адрес назначения, TCP порт приема, TCP-порт назначения, размер TCP окна приема, время прихода пакета.

Рис. 4.6. Алгоритм сбора сетевого трафика

По всем выборкам данных ведется обработка для каждого IP-адреса, и рассчитываются следующие характеристики трафика:

– суммарный объем данных, переданных за анализируемый интервал времени;

– средний интервал времени между передаваемыми пакетами.

Значения элементов массива структур и рассчитанные на их основе характеристики передаются на модуль формирования первичной статистики по http-трафику.

2) Алгоритм формирования статистики (рис. 4.7).

На втором этапе построения системы обнаружения Slow-HTTP атак, модуль формирования статистики извлекает набор записей о трафике за заданный интервал времени и выполняет расчет следующих показателей:

- количество сессий, за заданный интервал времени;
- объем данных по каждой сессии, с привязкой сессии к хосту;
- задержка между пакетами внутри сессии;
- скорость соединения.

Рис. 4.7. Алгоритм формирования статистики

Данный модуль фиксирует время начала и конца сессии, для отслеживания длительности открытых соединений.

3) Алгоритм расчета загрузки и перегрузки web-сервера, и принятия решения о наличие атаки (рис. 4.8).

На третьем этапе разработки системы обнаружения низкоинтенсивных атак типа «отказ в обслуживании» рассчитываются интенсивности поступления и обработки http-запросов на каждом промежутке времени: отношение количества поступивших пакетов с определенного IP-адреса к рассматриваемому промежутку времени.

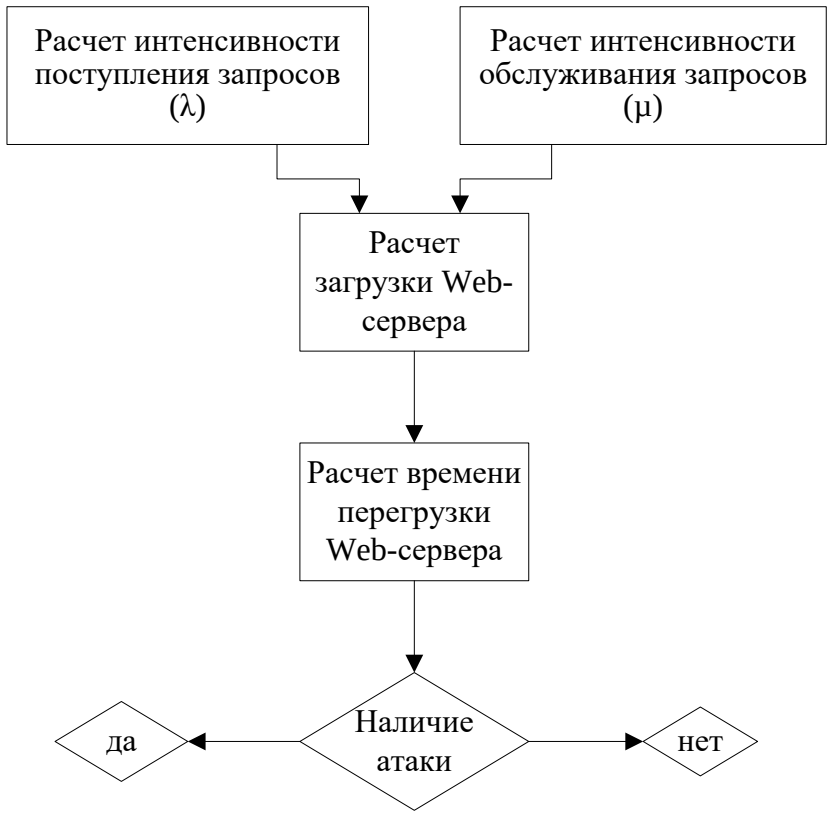


Рис. 4.8. Алгоритм расчета загрузки и перегрузки web-сервера, и принятия решения о наличие атаки

Интенсивность поступления http-запросов:

$$\lambda = \frac{k_i}{t}, \tag{4.5}$$

где:

λ- интенсивность поступления http-запросов;

k_i-количество поступивших запросов, анализируемых на данном промежутке;

t- рассматриваемый промежуток времени.

$$\mu = \frac{k_j}{t}, \tag{4.6}$$

где:

μ- интенсивность обработки http-запросов;

k_j-количество обработанных запросов, анализируемых на данном промежутке;

t- рассматриваемый промежуток времени.

Данные параметры соединения являются входными данными модуля, позволяющего рассчитать загрузку web-сервера на данный момент времени (4.2) и время его перегрузки (4.3).

Если отношение интенсивностей поступления запросов к их обработке стремится к значению максимального количества одновременно открытых соединений, прописанное в конфигурационном файле web-сервера «http.conf» , то вероятность Slowloris и Slow POST возрастает (рис. 4.9). Для исследуемого web-сервера значение «Max clients» равняется 150:

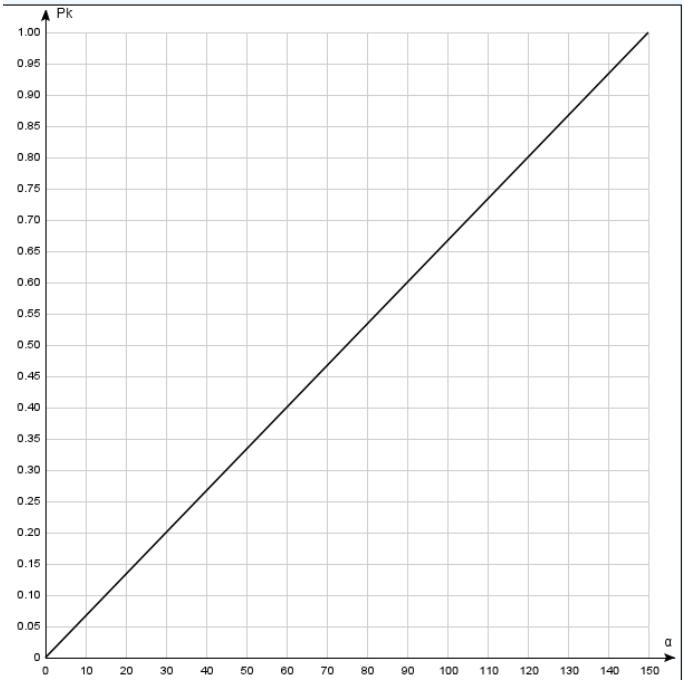


Рис. 4.9. График зависимости отношения поступивших запросов к обслуженным от вероятности возникновения Slowloris или Slow POST атак

Аналогично, если отношение интенсивностей обработанных заявок к поступившим стремится к значению максимального количества одновременно открытых соединений, то возрастает вероятность возникновения Slow READ (рис. 4.10).

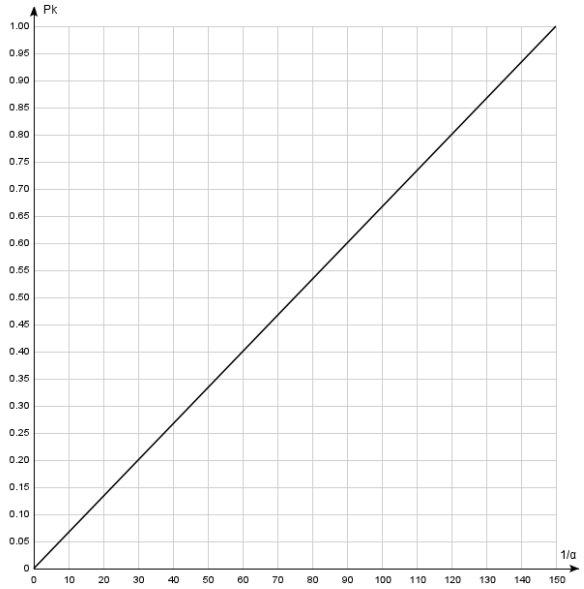


Рис. 4.10 График зависимости отношения обслуженных запросов к поступившим от вероятности возникновения Slowloris или Slow Post атак

Как отмечено в пункте 1.3, особенностью Slow HTTP атак является схожесть атакующего трафика с легальным, в случае загруженного канала связи, что составляет основную сложность их обнаружения. Предлагаемый алгоритм позволяет различать атакующий и легальный трафик, в часы максимальной интенсивности обращения пользователей к web-серверу, благодаря отслеживанию статистики соединения для каждого клиента: характерным параметром легального трафика в часы максимальной активности является большое количество открытых соединений с разных хостов, при постоянной частоте обслуживания запросов.

Что касается атакующего трафика, основной его характеристикой является множество открытых сессий с одного IP-адреса, время обслуживания которых стремится к значению тайм-аута соединения, прописанного в конфигурационном файле web-сервера "apache2.conf " и равному значению 300 сек, для исследуемого web-сервера.

На четвертом этапе, на основании рассчитанных параметров web-сервера, принимается решение о наличие атаки, в случае, если данные параметры не превышают статических порогов. Если параметры трафика не превышают заданных пороговых значений, принимается решение о его легальности, и трафик передается web-серверу. В случае превышения порогового значения,

принимается решение о возможной атаке, трафик с IP-адреса потенциального злоумышленника маркируется и его статистика передается на модуль классификации атак.

4.3.1. Классификация атакующего трафика.

В зависимости от параметров статистики соединения, поток запросов, направленный к web-серверу, может являться Slowloris атакой, Slow POST атакой или легальным трафиком. Обслуженные web-сервером запросы, направленные к клиенту в роли ответного трафика классифицируются как Slow READ атака либо как поток ответов на запросы легального пользователя.

Таким образом, модуль классификации атак должен решить задачу: отнести атакующий трафик к определенному классу Slow HTTP атак, либо снять маркер атаки, если параметры соединения не превысят пороговых значений.

Для решения данной задачи, разработано 2 алгоритма: алгоритм классификации Slow HTTP атак, для поступающего трафика и алгоритм классификации Slow HTTP атак, для ответного со стороны web-сервера потока трафика. Данные алгоритмы реализовываются соответственными модулями. Оба модуля работают в системе обнаружения атак параллельно, так как вероятность отказа в обслуживании web-сервера от исчерпания ресурсов поступающим потоком трафика, не зависит от вероятности отказа в обслуживании web-сервера от исчерпания ресурсов ответным со стороны web-сервера потока трафика.

Алгоритм распределения трафика по категориям атаки для поступающего потока трафика, в зависимости от параметров входящего соединения, изображен на рис. 4.11.

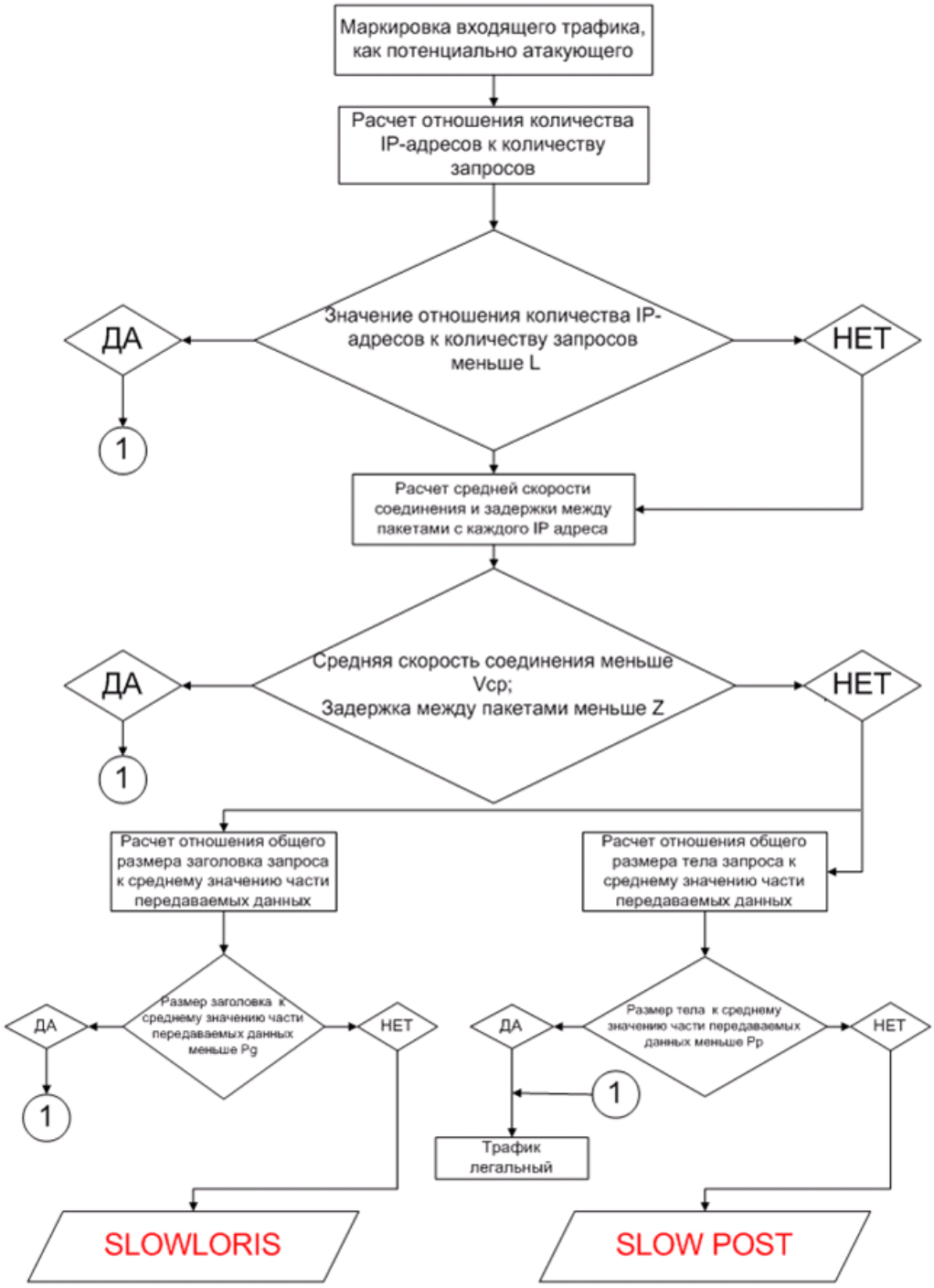


Рис. 4.11. Модуль классификации атак для поступающего потока трафика

1) На первом этапе рассчитывается отношение количества IP-адресов поступающего маркированного трафика, к количеству поступающих запросов за заданный интервал времени:

$$L(t) = \frac{N(t)}{k(t)}; \tag{4.9}$$

где:

$L(t)-$ отношение количества IP-адресов поступающего маркированного трафика, к количеству поступающих запросов за заданный интервал времени;

$N(t)$ – количество IP-адресов маркированного трафика;

$k(t)-$ количество запросов за заданный интервал времени.

Если значение $L(t)$ не превышает заданных порогов, то принимается решение о том, что поступающий трафик принадлежит легальному пользователю и маркер потенциально атакующего трафика снимается. Если же значение $L(t)$ превышает заданное пороговое значение, статистика соединения направляется на следующий этап классификации атак.

2) На втором этапе, для трафика, $L(t)$ которого превысило пороговое значение, рассчитывается средняя скорость соединения и задержка между пакетами для каждого IP-адреса:

$$V\text{ср} = \frac{\sum Vi}{i} \ , \tag{4.10}$$

где:

Vi- Скорость передачи i-ой части запроса;

i- количество передаваемых частей запроса.

$$Z = T_n - T_{n+1} \ , \tag{4.11}$$

где:

Z-задержка между пакетами;

T_n-время прихода n-ого пакета;

T_{n+1} –время прихода n+1 пакета.

Если значения $V\text{ср}$ и Z не превышают заданных порогов, то принимается решение о том, что поступающий трафик принадлежит легальному пользователю и маркер потенциально атакующего трафика снимается. Если же значение $V\text{ср}$ либо Z превышают заданное пороговое значение, статистика соединения направляется на следующий этап классификации атак.

3) На третьем этапе, для каждого метода запроса, рассчитываются отношения общего размера заголовка запроса к среднему значению части передаваемых данных за заданный промежуток времени для GET запросов, и отношения общего размера тела запроса к среднему значению части передаваемых данных за заданный промежуток времени для POST запросов.

$$Pg = \frac{Vg}{\sum Vg(i)/i} \ , \tag{4.13}$$

где:

Pg- отношение общего размера заголовка запроса к среднему значению части передаваемых данных за заданный промежуток времени для GET запросов;

Vg- общий размера заголовка запроса;

$Vg(i) -$ i-тая часть передаваемого заголовка запроса;

i-количество передаваемых частей заголовка запроса.

$$Pp = \frac{Vp}{\sum Vp(i)/i} \ , \tag{4.14}$$

где:

Pp- отношение общего размера тела запроса к среднему значению части передаваемых данных за заданный промежуток времени для POST запросов;

Vp- общий размера заголовка запроса;

$Vp(i) -$ i-тая часть передаваемого заголовка запроса;

i-количество передаваемых частей заголовка запроса.

Если значения Pg, для трафика с методом запроса GET либо Pp, для трафика с методом запроса POST не превышает заданных порогов, то принимается решение о том, что трафик принадлежит легальному пользователю и маркер потенциально атакующего трафика снимается.

Если же значение Pg превышает заданное пороговое значение, принимается решение об отнесении поступающего трафика к классу

низкоинтенсивных атак прикладного уровня Slowloris, и трафик с заданного IP-адреса блокируется.

Если же значение Pp превышает заданное пороговое значение, принимается решение об отнесении поступающего трафика к классу низкоинтенсивных атак прикладного уровня Slow POST, и трафик с заданного IP-адреса блокируется.

Алгоритм распределения трафика по категориям атаки для ответного со стороны web-сервера потока трафика, в зависимости от параметров входящего соединения, приведен на рис. 4.12.

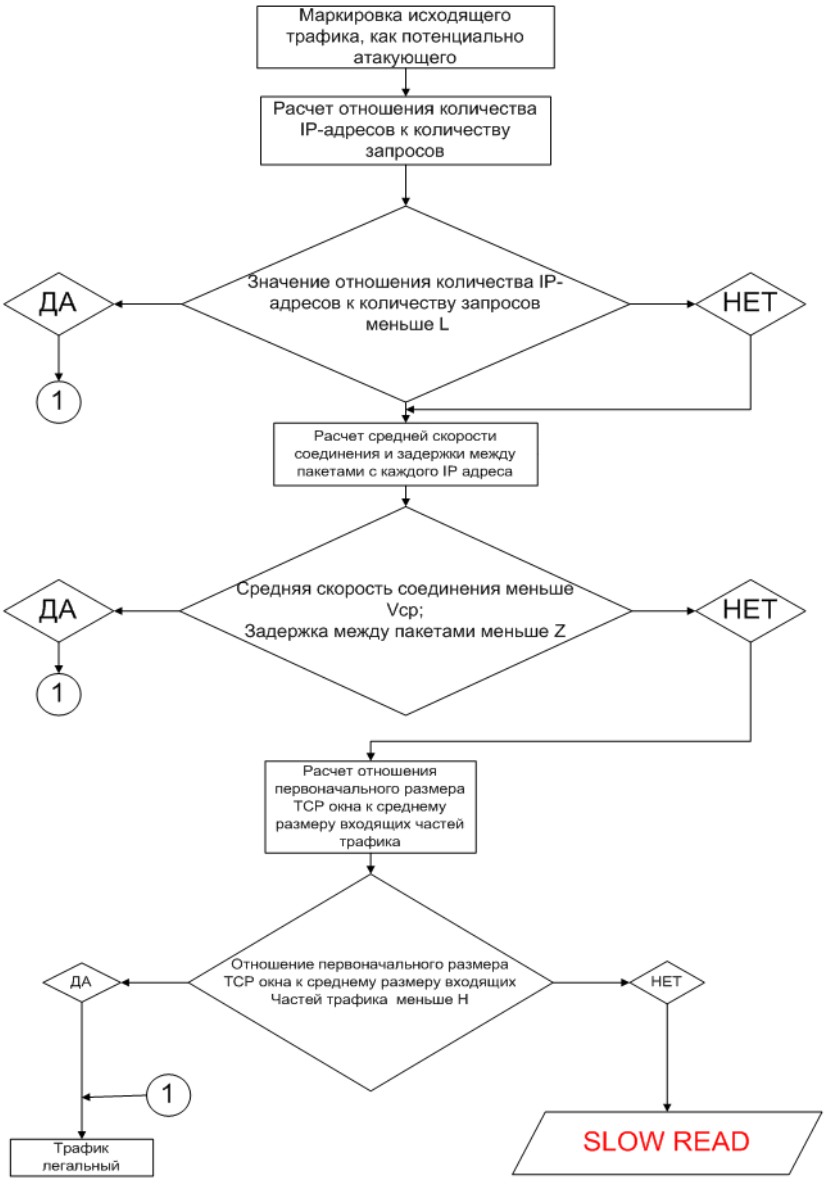


Рис. 4.12. Модуль классификации атак для ответного со стороны web-сервера потока трафика

1) На первом этапе рассчитывается отношение количества IP-адресов ответного со стороны web-сервера потока трафика, к количеству обслуживаемых запросов за заданный интервал времени (4.9).

Если значение $L(t)$ не превышает заданного порогового значения, то принимается решение о том, что ответный поток трафика принадлежит легальному пользователю и маркер потенциально атакующего трафика снимается.

Если же значение $L(t)$ превышает заданное пороговое значение, статистика соединения направляется на следующий этап классификации атак.

2) На втором этапе, для трафика, $L(t)$ которого превысило пороговое значение, рассчитывается средняя скорость соединения и задержка между ответными пакетами для каждого IP-адреса (4.10, 4.11).

Если значения $V\text{ср}$ и Z не превышают заданных пороговых значений, то принимается решение о том, что трафик принадлежит легальному пользователю и маркер потенциально атакующего трафика снимается. Если же $V\text{ср}$ либо Z превышают заданное пороговое значение, статистика соединения ответного потока трафика направляется на следующий этап классификации атак.

3) На третьем этапе, рассчитывается отношение первоначального размера TCP окна приема, к размеру частей потока ответного трафика:

$$H = \frac{Vtcp}{\sum V(i)/i}\ , \qquad (4.15)$$

где:

H - отношение первоначального размера TCP окна приема, к размеру частей потока ответного трафика;

$Vtcp$ - первоначальный размер TCP окна приема;

$V(i) -\ i$-тая часть ответного запроса;

i-количество обслуживаемых частей запроса за заданный интервал времени.

Если значение H ответного со стороны web-сервера потока трафика не превышает заданных порогов, принимается решение о том, что ответный трафик

принадлежит легальному пользователю и маркер потенциально атакующего трафика снимается.

Если же значение H превышает заданное пороговое значение, принимается решение об отнесении трафика к классу низкоинтенсивных атак прикладного уровня Slow READ и трафик с заданного IP-адреса блокируется.

4.4. Анализ эффективности метода обнаружения и классификации Slow-HTTP атак.

Для оценки эффективности разработанного метода и системы обнаружения slow-HTTP атак на облачную инфраструктуру выполненно ряд экспериментов.

На рис. 4.13 приведена общая структурная схема топологии облачной инфраструктуры на базе технологии OpenStack.

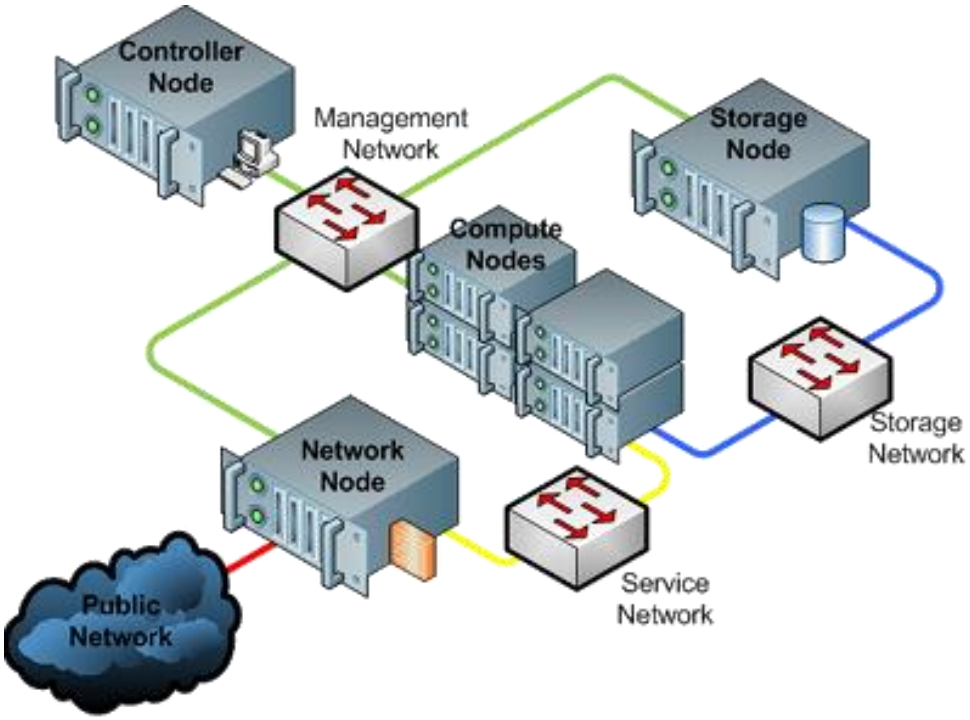


Рис. 4.13. Схема топологии развертывания OpenStack

На базе данной технологии была развернута виртуальная лаборатория ReSeLA, разработанная в рамках проекта TEMPUS ENGENSEC. Эта лаборатория, развернутая в облачной инфраструктуре позволяет изучать

вопросы связанные с информационной безопасностью (обнаружение вторжений, анализ уязвимостей, анализ зловредного программного обеспечения и др.).

Обеспечение изолированного окружения для проведения исследований в области информационной безопасности имеет решающее значение, когда экспериментирование с самовоспроизводящимся кодом, например. Вирусами не допускается к распространению за пределами лаборатории.

Платформы виртуализации могут использовать аппаратные ресурсы лучше, чем эмулированные платформы, и, безусловно, будет дешевле в настройке и запуске, в зависимости от того, какое оборудование и используемая программная платформа. Запуск виртуального Машина (VM) работает быстрее, чем физическая запись на жесткий диск, поэтому время запуска эксперимента будет короче. По окончании эксперимента содержимое диска виртуальных машин может быть отбрасывается, что означает, что вам не нужно вытирать жесткий диск. Также будет легче подготовить установки, поскольку учитель может это сделать на своей машине в виртуализированном окружающей среды, не беспокоясь о несовместимости оборудования. С другой стороны, использование виртуальных машин может привести к нежелательным поведение. Если многие эксперименты будут работать на одном и том же машине, они должны совместно использовать аппаратные ресурсы между друг друга, что означает, что они также влияют друг на друга. Эксперименты, которые являются ресурсоемкими, занимают больше времени выполнить и результаты измерений производительности не будут всегда быть надежным. Конкуренция на ресурсах также может влиять результаты экспериментов в зависимости от параллелизма, один пример из домена безопасности - условия гонки.

Запуск нескольких виртуальных сред на одном компьютере также приведет к штрафам за доступ к жесткому диску. В виртуализированной среде, в которой эмулируется аппаратное обеспечение. Прямой доступ к оборудованию может быть сделано в форме делегирования устройства, однако это механизм не хорошо автоматизирован в существующих IaaS.

Эмуляция может ввести новые ошибки связанные с программным обеспечением эмуляции и, вероятно, скрыть ошибки в реальном оборудовании.

Эксперименты, требующие прямого аппаратного обеспечения таким образом, доступ к нему будет невозможным в первой версии ReSeLa, как и эксперименты в зависимости от аппаратных ошибок.

Архитектура лаборатории приведена на рис. 4.14. Функциональная архитектура – рис. 4.15.

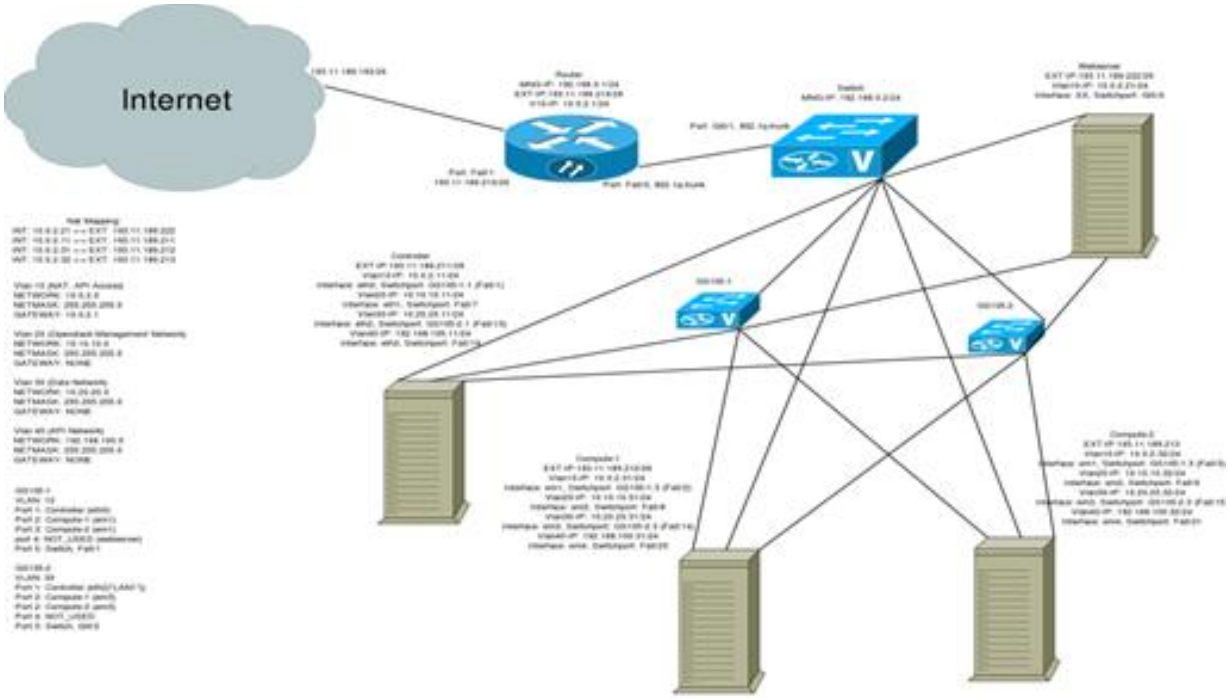


Рис. 4.14. Обобщенная архитектурная схема RESELA

Анализ архитектуры облачной среды и виртуальной лаборатории показал, что она содержит 2 сети: внутреннюю и внешнюю. Все сетевые интерфейсы, ассоциированные со внешней сетью соответствуют инстрансам виртуальных машин, к которым должен быть предоставлен из Internet, включая Web-траффик. Внутренняя сеть используется для взаимодействия между web-серверами отдельных компонентов системы.

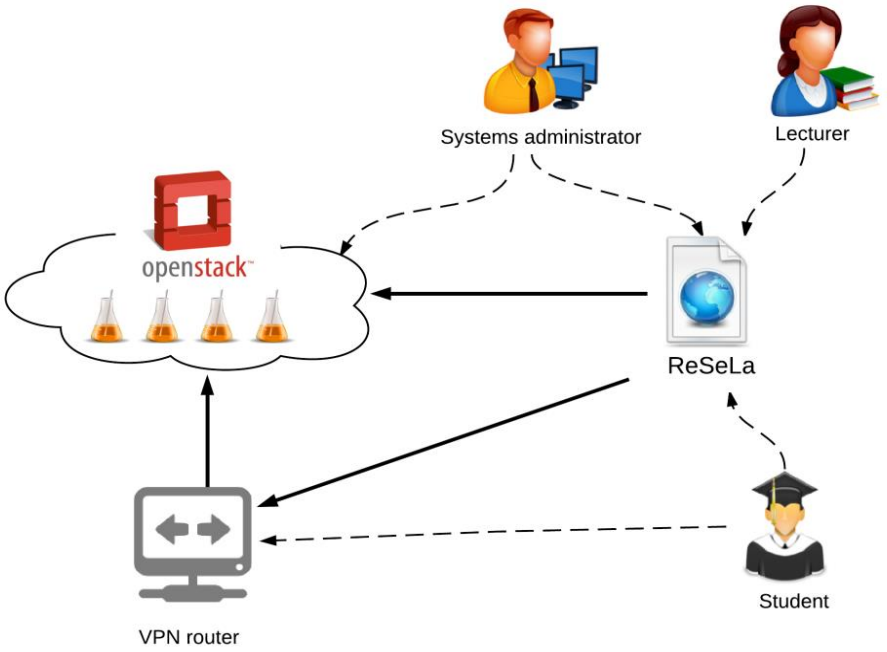


Рис. 4.15. Функциональна схема RESELA

На данную лабораторию моделировался поток http-запросов в соответствии с алгоритмами slow-http атак. Общая структурная схема эксперимента атаки приведена на рис. 4.16.

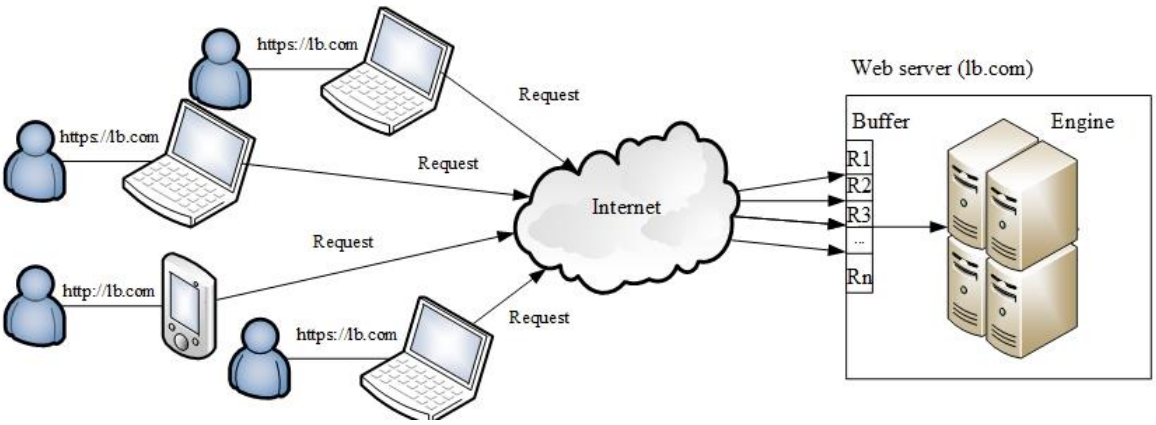


Рис. 4.16. Обобщенная схема slow-http атаки на облачную инфраструктуру

Так же на рис. 4.17-4.20 приведены сценарии атак на отдельные компоненты облачной инфраструктуры.

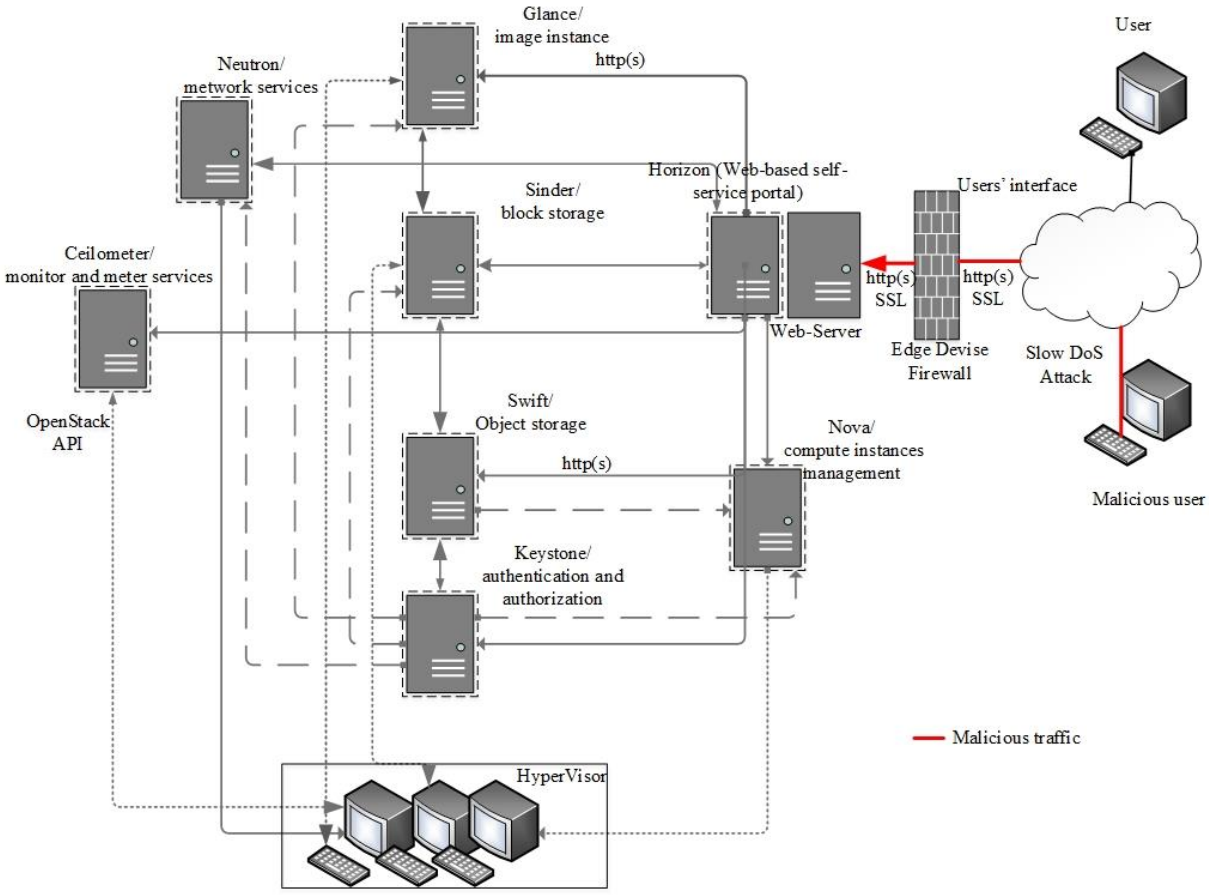


Рис. 4.17. Сценарий slow-http атаки на Horizon модуль

Реализация сценария на slow-http атаки на модуль Horizon заключалась в реализации атаки Slowloris, когда формируется большое количество открытых соединений путем непрерывной отправки незавершенных GET-запросов. Данная атака маскируется под ситуацию роста популярности облачного сервиса и получения большого количества запросов на обслуживание. В тоже время открытые соединения не завершаются.

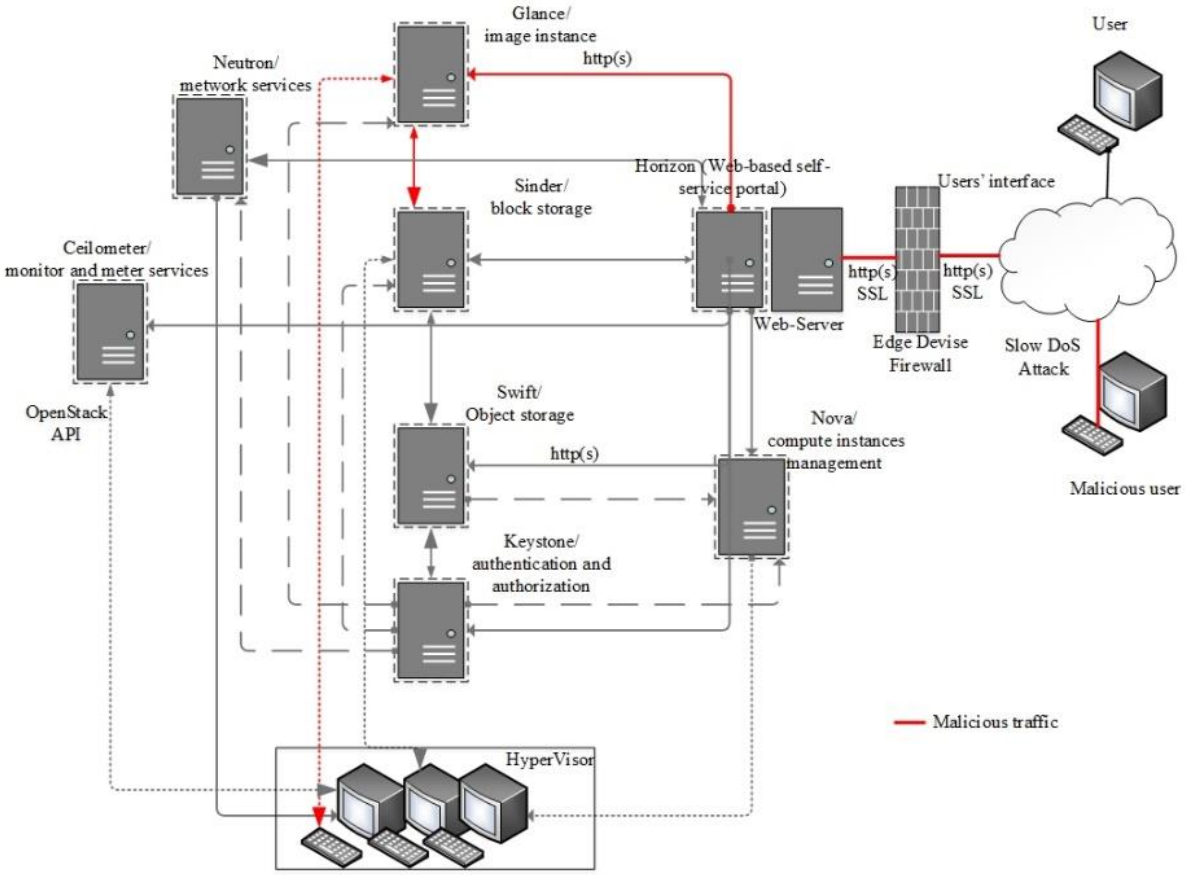


Рис. 4.17. Сценарий slow-http атаки на Glance модуль

Сценарий атаки на Glance модуль заключался в создании образов нулевого размера ('glance image-create' – метод вызывается без параметров). Это тип атаки реализуется внутри системы OpenStack. Атакующий может отправить запрос путем модификации оригинального POST запроса на создание образа.

Скрипт атаки включает в себя команду:

*i -X POST -H 'X-Auth-Token: ****

Сценарий атаки на модуль Keystone основан на реализации Slow HTTP POST атаки: формировался POST заголовок с легитимным полем «Content-Length», которое позволяет web-серверу идентифицировать объём данных, который к нему поступает. Как только заголовок отправлен, тело POST запроса начинало передаваться с очень медленной скоростью, что не давало web-серверу возможности обрабатывать другие запросы.

В каждом из приведенных сценариев полученный http request не обрабатывается до конца или передача данных выполняется с очень низкой скоростью (малый размер окна TCP и задержки между пакетами).

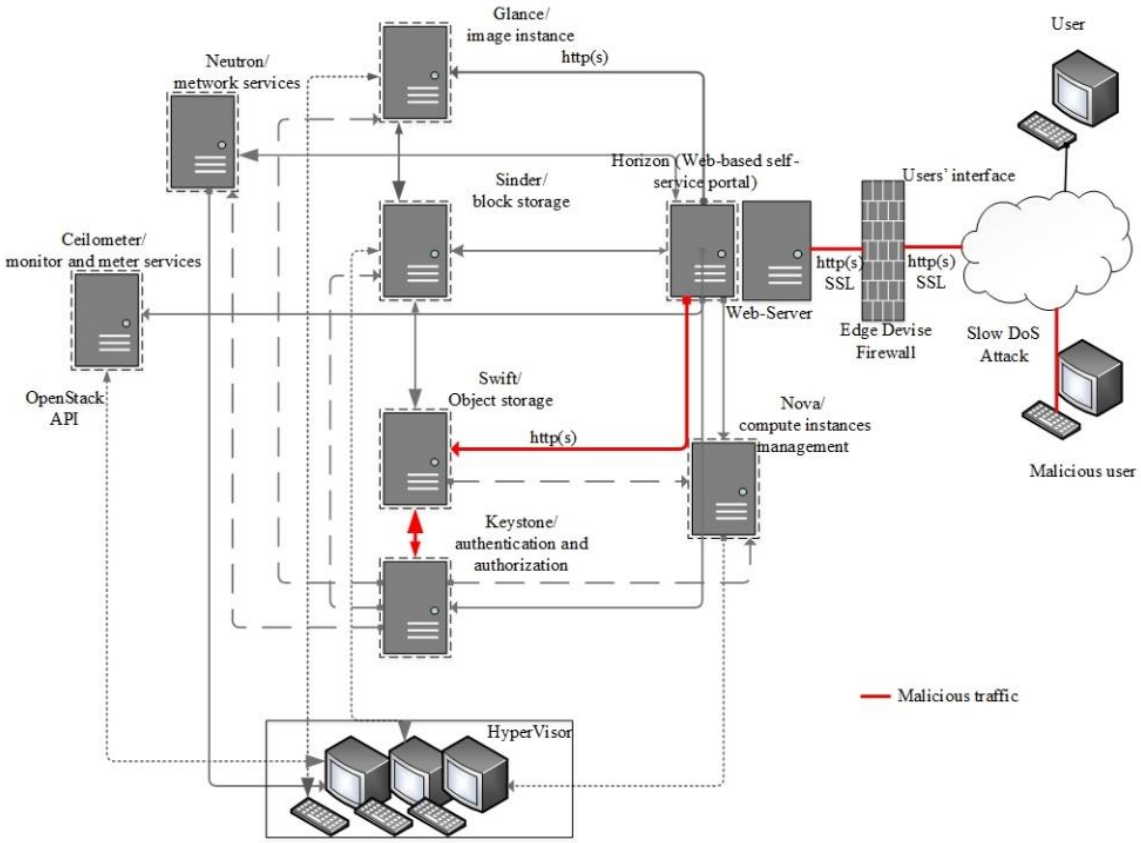


Рис. 4.18. Сценарий slow-http атаки на Keystone модуль

Результаты проведения атак по различным сценариям приведены на рис. 4.18-4.20.

Как видно из приведённой диаграммы на первом этапе реализации slow-http атаки было сформировано большое количество (1000) соединений, часть из них была принята в обработку и данные соединения помечены как Connected, остальные соединения поставлены в очередь на обслуживание (Pending). Уже на 10-й секунде был зафиксирован первый «отказ в обслуживании» для запроса не принадлежащего атакующему трафику.

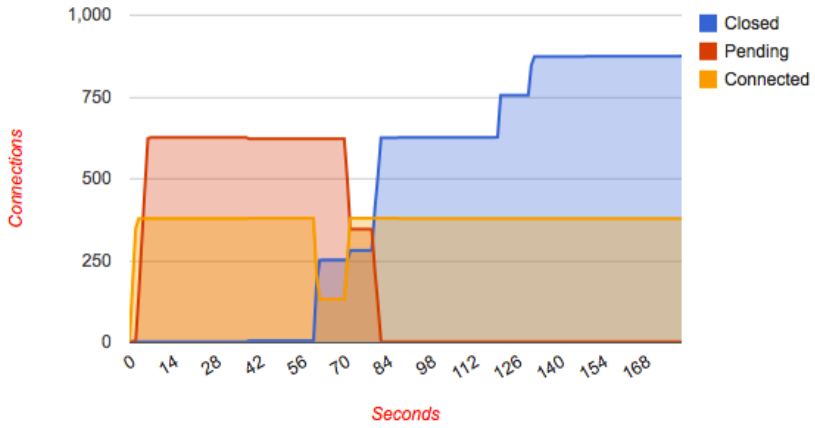


Рис. 4.18. Диаграмма соединений при реализации slow-http атаки на модуль Horizon

После включения системы обнаружения slow-http атак начал выполняться отброс подозрительных соединений, что показывает синяя линия графика (Closed). Это позволило очистить очередь входящих соединений от атакующего трафика и вернуть сервис в состояние доступности к обслуживанию.

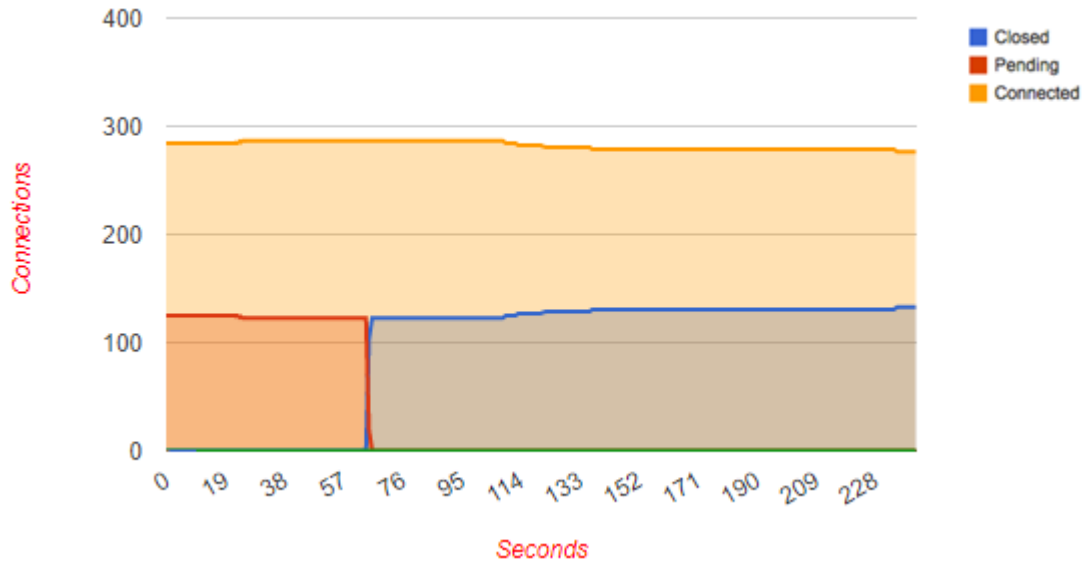


Рис. 4.19. Диаграмма соединений при реализации slow-http атаки на модуль Glance

Как было сказано выше атака на модуль Glance заключалась в генерации потока POST запросов на формирование новых образов виртуальных машин, без указания их объема. Как видно из диаграммы после включения системы

обнаружения slow-http атак удалось очистить буфер входящих запросов от атакующего трафика.

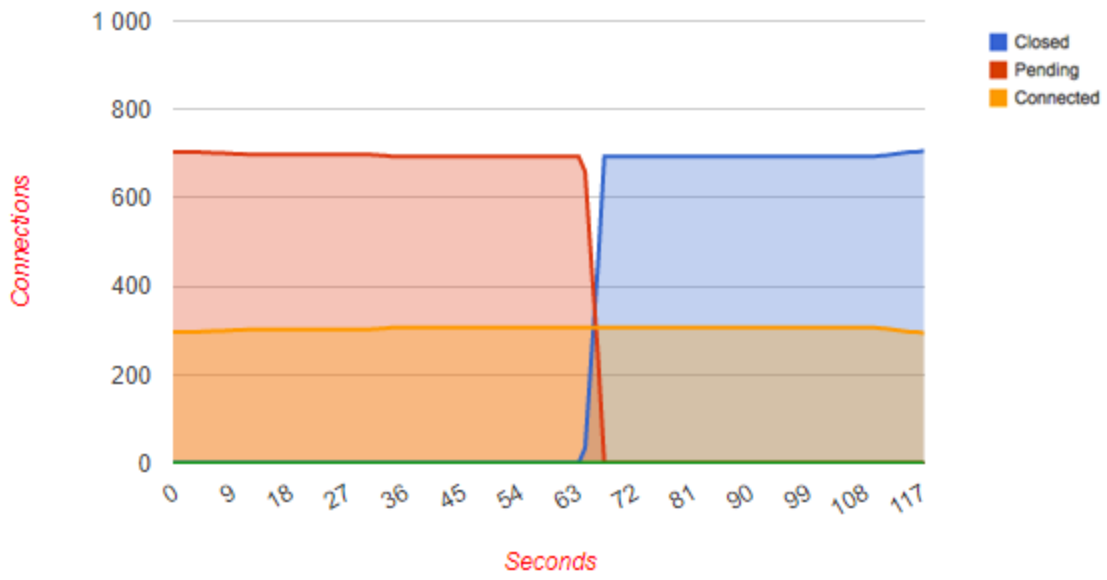


Рис. 4.20. Диаграмма соединений при реализации slow-http атаки на модуль Keystone

Модуль Keystone является самым нагруженным модулем системы OpenStack, поэтому, как правило для поддержи данного модуля используется несколько параллельно работающих web-серверов. Использование горизонтального масштабирования позволяет динамически увеличивать мощность системы обработки, но в при реализации slow-http атаки не позволяет полностью защитить систему от перехода в состояние «отказ в обслуживании». При проведении эксперимента использовалось 2 web-сервера с возможностью одновременного обслуживания по 128 соединений каждый. С точки зрения реализации атаки это просто увеличило время на заполнение всего пула входных запросов. Следовательно, для систем, которые используют горизонтальное масштабирование web-серверов так же необходимы средства защиты.

Подключение системы обнаружения slow-http атак позволило отсеять зловредный трафик и выйти из состояния «отказ в обслуживании».

Анализ данных графиков показывает, что использование предлагаемого подхода позволятет отсекать подозрительные сессии к атакуемому web-серверу, тем самым освобождая ресурсы для обслуживаня легальных http-запросов.

Для противодействия slow-http атакам предлагается два основных подхода:

– конфигурирование правил межсетевых экранов;

–конфигурирование систем обнаружения вторжений (IDS).

Конфигурирование правил межсетевых экранов выполняется путем определения статических или динамических правил блокировки трафика.

1) Статический лимит соединений

```
iptables -I INPUT -p tcp --syn --dport 80 -m connlimit \
--connlimit-above 5 --connlimit-mask 32 -j DROP
iptables -I INPUT -p tcp --dport 80 -j ACCEPT
```

Например данные правила ограничивают число соединений пятью с одного узла. При превышении этого лимита попытка соединения запрещается. Если у атакующего достаточно большой ботнет, он всё равно может успешно атаковать. Также значение connlimit должно быть очень низким, чтобы защитить web-сервер. Это может существенно затруднить использование сервера легитимными клиентами и вызывает большое число ложных срабатываний (false positives). Такие правила затрудняют или исключают использование NAT и proxy-серверов.

2) Динамический лимит соединений

```
iptables -I INPUT -p tcp -m state --state NEW --dport 80\
-m recent --name slowloris --set
iptables -I INPUT -p tcp -m state --state NEW --dport 80\
-m recent --name slowloris --update \
--seconds 15 --hitcount 10 -j DROP
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

Первые два правила проверяют новые соединения и число соединений с одного ip-адреса. Если с одного ip–адреса создаётся более 10 соединений в течение 15 секунд, то такие пакеты блокируется. Для такого набора правил

характерно неприемлемо большое число ложных срабатываний [23]. Легитимные подключения хостов через один NAT и proxy-сервер также будут удовлетворять этим правилам и будут блокироваться межсетевым экраном.

Конфигурирование правил IDS, например системы Snort, входящие в состав набора правил по умолчанию [24], основано на поиске строковых шаблонов, специфичных для определённых сценариев атак, и контроле пороговых значений пакетов в единицу времени.

```
alert tcp any any -> any any (msg:"low-rate DDoS";
flow:to_server,established; content:”some DoS tool user-agent specific content”
detection_filter:track by_src, count 20, seconds 20;
metadata:service http; classtype:attempted-dos; sid:1234572; rev:2;)
```

4.5. Выводы по разделу.

Анализ облачной инфраструктуры на базе OpenStack позволил выявить наиболее уязвимые компоненты для реализации DOS-атак в частности low-http атак.

Для повышения защищенности облачной инфраструктуры предложена разработка системы обнаружения и классификации slow-http атак.

Разработанные модели обнаружения slow-http атаки и прогнозирования времени перехода web-сервера в состояние «отказ в обслуживании» использованы в качестве основы для системы обнаружения и предотвращения slow-http атак.

Эффективность разработанной системы оценивалась путем проведения эксперимента по реализации slow-http атаки на виртуальную лабораторию ReSeLa.

Полученные результаты показали, что использование предложенных моделей обнаружения slow-http атак и прогнозирования времени перехода в состояние «отказ в обслуживании» позволили повысить доступность сервера на 50-75% для различных типов slow-http атак.

ВИСНОВКИ

В роботі розв'язана ***актуальна науково-прикладна задача,*** що полягає у підвищенні доступності послуг, що надаються мультисервісною мережею за рахунок вдосконалення існуючих та розробки нового методу виявлення мережевих атак.

За підсумками вирішення науково-прикладної задачі зроблено наступні висновки:

1. Аналіз поточного стану та перспектив розвитку мультисервісних мереж показав, що все частіше причиною відсутності доступу до певних сервісів є мережеві атаки. В останній час все більшої популярності набувають так звані атаки на «відмову в обслуговуванні», що реалізуються на прикладному рівні.

2. Проведений аналіз методів та засобів виявлення та протидії мережевим атакам на «відмову в обслуговуванні» показав, що основні зусилля існуючих засобів націлено на виявлення та протидію атакам, що реалізовано на мережевому та канальному рівнях. Так, загально відомі засоби та методи виявлення атак на відмову в обслуговуванні базуються на детектуванні аномалій трафіку, характерних для атак. У той же час, атаки на відмову в обслуговуванні прикладного рівня не вимагають генерації великого обсягу трафіку, тому атаки цього типу досить важно ідентифікуються розповсюдженими методами та засобами виявлення та протидії мережевим атакам на «відмову в обслуговуванні».

3. Проведено експеримент з реалізації низькоінтенсивних атак прикладного рівня на web-сервер (slow-http). В ході виконання експерименту досліджено різні сценарії даного типу атак, виявлено відмінні риси та параметри трафіку, що можуть ідентифікувати кожний тип атаки.

4. Розроблено модель поведінки web-серверу, що дозволяє визначити імовірність реалізації slow-http атаки. Модель базується на використанні ланцюгів Маркова та дозволяє розрахувати розподіл імовірностей між можливими станами web-серверу.

5. Розроблено модель визначення часу переходу web-серверу у стан «відмова в обслуговуванні». Модель базується на використанні ймовірносно-

часових функцій та базується на інформації, що попередньо отримана за допомогою моделі станів web-серверу.

6. Проведено експеримент з моделювання атак на web-сервер, що дозволив перевірити адекватність розроблених моделей.

7. Розроблено загальний метод виявлення та класифікації атак на відмову в обслуговуванні прикладного рівня на web-сервер. Розроблений метод дозволяє виявити факт виконання атаки, визначити джерело нападу та блокувати зловмисний трафік.

8. В основі запропонованої системи захисту лежить аналіз поведінки сервера в нормальному режимі та при реалізації різновидів типу Slow-http атак. Аналіз дозволив ідентифікувати найбільш чутливі параметри для атак повільного нападу, а також встановити їх пороги в залежності від потужності каналу, продуктивності обладнання та параметрів конфігурації сервера.

9. Процес виявлення атаки реалізується на основі моделі Маркова за поведінкою сервера, параметрами моделі є статистичні характеристики вхідного, вихідного трафіку, а також динаміки серверу ресурсного використання. Перевага запропонованої системи полягає в тому, що вона дозволяє виявляти атаку на відмову в обслуговуванні сервера, що дає можливість своєчасно активувати відповідні механізми захисту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Anders Carlsson ИНФРАСТРУКТУРА PenTestING И УПРАВЛЕНИЯ УЯЗВИМОСТЬЮ/ Хаханов Владимир Иванович, Чумаченко Светлана Викторовна, Anders Carlsson // АСУ и приборы автоматики. – 2012. – №160 – С. 36-51
2. Anders Carlsson Модели управления уязвимостью /Хаханов Владимир Иванович, Anders Carlsson, Чумаченко Светлана Викторовна, Бутенко Сергей Александрович // АСУ и приборы автоматики. – 2012. – №161. – С.10-24
3. Carlsson A. Analysis of realization and method of detecting low-intensity HTTP-attacks [Электроний ресурс] / A. Carlsson, E.V. Duravkin, A.S. Loktionova // Проблеми телекомунікацій. – 2013. – № 3 (12). – С. 61-70. – Режим доступу до журналу: http://pt.journal.kh.ua/2013/3/1/133_carlsson_ attack.pdf
4. Carlsson A. A. Analysis of realization and method of detecting low-intensity HTTP-attacks. Part 2. Method of detecting Slow HTTP attacks [Электроний ресурс] / A.A. Carlsson, I.V. Duravkin, A.S. Loktionova // Проблеми телекомунікацій. – 2014. – № 1 (13). – С. 96-100. – Режим доступу до журналу: http://pt.journal.kh.ua/2014/1/1/ 141_carlsson_attack.pdf.
5. Carlsson A. A. Method of slow-attack detection / Carlsson Anders, I. V. Duravkin, A. S. Loktionova // Системи обробки інформації. — 2014. — № 8. — С. 102-106.
6. Carlsson Anders. Detecting cyber threats through social network analysis/ Carlsson Anders, Kirichenko Lyudmyla, Radivilova Tamara // SocioEconomic Challenges, – №1(1), – 2017. –p. 20-34.
7. Anders Carlsson. Model of network attack on the cloud platform OpenStack/ Anders Carlsson // In proc. of Second International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T'2015), Kharkiv, Ukraine. – 2015. – p. 30-33
8. Anders Carlsson Infrastructure of pentesting and vulnerability management // Vladimir Hahanov, Anders Carlsson, Svetlana Chumachenko/ In Proc. of conference, Kharkov, Ukraine,ISBN 0135-17, 10 Sep. 2012, – P. 10-24.
9. Wajeb Gharibi, Hahanov V. I., Anders Carlsson, Hahanova I. V., Filippenko I.V. Quantum technology for analysis and testing computing systems // In Proc. of IEEE EastWest Design & Test Symposium (EWDTS'2013), Rostov-on-Don, Russia, Sep. 27-30 , 2013.
10. Anders Carlsson; Rune Gustavsson: "Resilient Smart Grids."// In Proc. of First International Scientific-Practical Conference Problems of Infocommunications Science and Technology, Kharkov, Ukraine, Oct. 14-17, 2014
11. Alexander Adamov, Vladimir Hahanov, Anders Carlsson. Discovering New Indicators for Botnet Traffic Detection / Alexander Adamov, Vladimir Hahanov,

Anders Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2014), September 26–29, 2014, Kiev, Ukraine. – Kiev, 2014. – P. 281–285.

12. Anders Carlsson, Rune Gustavsson. Resilient Smart Grids //In Proc. of First International Scientific-Practical Conference Problems of Infocommunications Science and Technology, Kharkov, Ukraine, Oct. 14-17, 2014 PIC_S&T– pp. 79-82

13. Adamov A, Carlsson A. A Sandboxing Method to Protect Cloud Cyberspace / Adamov A, Carlsson A // Proceedings of IEEE East-West Design & Test Test Symposium (EWDTS'2015), September 27-30, 2015, Batumi, Georgia – P. 180–183.

14. Anders Carlsson. Model of network attack on the cloud platform OpenStack / Anders Carlsson // In proc. of Second International Scientific-Practical Conference Problems of Infocommunications Science and Technology (PIC S&T'2015), Kharkiv, Ukraine, Oct. 13 -15, – 2015

15. Kurt Tutschku, Vida Ahmadi Mehri, Anders Carlsson, Krishna Varaynya Chiukula, Johan Christenson. On Resource Description Capabilities of On-Board Tools for Resource Management in Cloud Networking and NFV Infrastructures / Kurt Tutschku, Vida Ahmadi Mehri, Anders Carlsson, Krishna Varaynya Chiukula, Johan Christenson // In Proc. of First IEEE International Workshop on Orchestration for Software Defined Infrastructures (co-located with IEEE ICC 2016), Kuala Lumpur, Malaysia, May 23 - 27, 2016.

16. Kurt Tutschku, Vida Ahmadi Mehri, Anders Carlsson. Towards Multi-layer Resource Management in Cloud Networking and NFV Infrastructures / Kurt Tutschku, Vida Ahmadi Mehri, Anders Carlsson // In Proc. of 12th Swedish National Computer Networking Workshop (SNCNW), Sundsvall, Sweden, Jun. 1-2, 2016.

17. Alexander Adamov, Anders Carlsson. Cloud incident response model / Alexander Adamov, Anders Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2016), October 14–17, 2016, Yerevan, Armenia. – P. 250–253.

18. Ievgeniia Kuzminykh, Arkadii Snihurov, Anders Carlsson. Testing of communication range in ZigBee technology / Ievgeniia Kuzminykh, Arkadii Snihurov, Anders Carlsson // In Proc. of 14th International Conference on The Experience of Designing and Application Systems in Microelectronics (CADSM'2017), Polyana, Svalyava (Zakarpattya), Ukraine, Feb. 21 – 25, 2017.

19. Alexander Adamov, Anders Carlsson. The State of Ransomware. Trends and Mitigation Techniques / Alexander Adamov, Anders Carlsson // Proc. of IEEE East-West Design & Test Symposium (EWDTS'2017), Sep 29 – Oct 2, 2017, Belgrad, Serbia. – P. 121–128.

20. Исследование операций: в 2-х томах // Пер. с англ. / Под ред. Дж. Моуде-ра, С. Элмаграби. -М.: Мир, 1984. Т. 1. - 712 с.

21. Исследование операций: в 2-х томах // Пер. с англ. / Под ред. Дж. Моуде-ра, С. Элмаграби. М.: Мир, 1984. - Т.2. - 677 с.

22. Кобринский Н.Е., Майминас Е.З., Смирнов А.Ф. Введение в экономическую кибернетику. М.: Экономика, 1975.

23. Грэм Р.Г., Грей К.Ф. Руководство по операционным играм // Пер. с англ. / Под ред. Широкова Ф.В. М.: Сов. радио, 1977. - 376 с.

24. Кини P.JL, Райфа Х. Принятие решений при многих критериях: предпочтения и замещения // Пер. с англ. / Под ред. Шахнова И.Ф. М.: Радио и связь, 1981.-560 с.

25. Теория выбора и принятия решений: Учебное пособие. М.: Наука, 1982. -328 с.

26. Орловский С.А. Проблемы принятия решений при нечеткой исходной информации. М.: Наука, 1981. - 208 с.

27. Моисеев Н.Н. Неформальные процедуры и автоматизация проектирования / Новое в жизни, науке, технике: Математика, кибернетика. М.: Знание, 1979.

28. Царегородцев А.В. Современные технологии управление в человеко-машинных системах. М.: Радио и связь, 2002. - 184 с.

29. Царегородцев А.В. Основы теории построения платформ безопасности интегрированных производственных комплексов. М.: Изд-во Физико-математической литературы, 2003. - 184 с.

30. Шаракшанэ А.С. и др. Сложные системы: Уч. пособие для вузов. М.: Высшая школа, 1977. - 247 с.

31. Вилкас Э.Ш, Майминас Е.З. Теория, информация, моделирование. М.: Радио и связь, 1981.-327 с.

32. Флейшман Б.С. Основы системологии. М: Радио и связь, 1982. - 368 с.

33. Флейшман Б.С., Брусиловский П.М. О методах математического моделирования систем. В кн.: Системные исследования. - М: Радио и связь, 1983.

34. Поспелов Д.А., Пушкин В.Н. Мышление и автоматы. М.: Сов. радио, 1972.-223 с.

35. Поспелов Д.А. Логико-лингвистические модели в системах управления. -М: Энергоиздат, 1981. 232 с.

36. Клыков Ю.И. Ситуационное управление большими системами. М.: Энергия, 1974.- 136 с.

37. Бурков В.Н., Кондратьев В.В. Механизмы функционирования организационных систем. -М.: Наука, 1981.

38. Кондратьев В.В. и др. Теория активных систем и совершенствование хозяйственного механизма. М.: Наука, 1984.

39. Горбатов В.А. Теория частично-упорядоченных систем. М.: Сов. радио, 1976.-336 с.

40. Любищев А.А. Значение и будущее систематики. М.: Природа, 1977. - №2.-С. 112-145.

41. Сагатовский В.Н. Системная деятельность и ее философское осмысление.- В кн.: Системные исследования. Ежегодник, 1980. М.: Наука, 1981. - С. 52-68.

42. Битунов В.В., Чистяков Е.Г., Шульга В.А. Методы планирования хозяйства города. М.: Экономика, 1981.

43. Корпоративные сети и системы. Экономические данные // Компьютерра.- 1997.-№42.

44. Темп прироста рынка ИТ в Европе достиг двузначной цифры // Сети и системы связи. Ноябрь. - 1999.45.0тоцкий Л., Савин А. Тернистый путь к современной технологии управления // Открытые системы. 1998. - № 2.

45. Ладыженский Г.М. Архитектура корпоративных информационных систем //СУБД. -1997.- №5, 6.47.3индер Е.З. Соотнесение и использование стандартов организации жизненных циклов систем // СУБД. 1997. - № 3.

46. Вендров А.М. CASE -технологии. Современные методы и средства проектирования информационных систем. -М.: Финансы и статистика, 1998.

47. Международные стандарты, поддерживающие жизненный цикл программных средств. М.: МП «Экономика», 1996.

48. Ахтырченко К.В., Леонтьев В.В. Распределенные объектные технологии в информационных системах // СУБД. 1997. - № 5,6.

49. Девянин П.Н., Михальский О.О., Правиков Д.И., Щербаков А.Ю. Теоретические основы компьютерной безопасности. М.: Радио и связь, 2000.

50. Пупков К.А., Ломакин И.В. Опыт разработки и создания систем обеспечения управленческих решений. В сб.: Вопросы информационной технологии. -М.: ВНИИСИ, 1982. - Вып.1. - С. 42-52.

51. Царегородцев А.В. Информационная безопасность в распределенных управляющих системах. М.: Изд-во РУДН, 2003. - 220 с.

52. Анохин П.К. Очерки по физиологии функциональных систем. М.: Медицина, 1975.-448 с.

53. Месарович М., Мако Д., Такахара И. Теория иерархических многоуровневых систем. М.: Мир, 1973. - 343 с.

54. Общая теория систем / Пер. с англ. М.: Мир, 1973. - 343 с.

55. Беллман Р., Заде Л. Принятие решений в расплывчатых условиях. В сб.: Вопросы анализа и процедуры принятия решений. - М.: Мир, 1976. - С. 175-215.

56. Акофф Р., Эмери Ф. О целеустремленных системах / Пер. с англ. М.: Сов. радио, 1974. - 272 с.

57. Черняк Ю.И. Системный анализ в управлении экономикой. М.: Экономика, 1975.

58. Системный анализ и структуры управления / Под общей ред. проф. В.Г. Шорина. М.: Знание, 1975. - 303 с.

59. Дружинин В.В., Конторов Д.С. Проблемы системологии (проблемы теории сложных систем). М.: Сов. радио, 1976. - 296 с.

60. Багриновский К.А. Основы согласования плановых решений. М.: Наука,1977.-303 с.

61. Багриновский К.А., Бусыгин В.П. Математика плановых решений. М.: Наука, 1977.-234 с.

62. Кон П. Универсальная алгебра / Пер. с англ. М.: Мир, 1968. - 351 с.

63. Курош А.Г. Общая алгебра. М.: Наука, 1974.

64. Кейслер Г., Чэн Ч.М. Теория моделей // Пер. с англ. / Под ред. Ершова Ю.Л. и Тайманова А.Д. М.: Мир, 1977. - 614 с.

65. Сакс Дж. Е. Теория насыщенных моделей // Пер. с англ. / Под ред. А.Д. Тайманова. -М.: Мир, 1976. 196 с.

66. Николаев Ю.И. Проектирование защищенных информационных технологий. С.-Пб.: Изд-во СпбГТУ, 1997.

67. Мостовскии А. Конструктивные множества и их приложения // Пер. с англ. / Под ред. Тайманова А.Д. М.: Мир, 1973. - 256 с.

68. Садовский В.Н. Проблемы философского обоснования системных исследований. В кн.: Системные исследования. Ежегодник, 1984. - М.: Наука, 1984. -С. 32-51.

69. Лапшинский А.В. Локальные сети персональных компьютеров. М.: МИФИ, 1994.

70. Садовский В.Н. Принцип системности, системный подход и общая теория систем. В кн.: Системные исследования. Ежегодник, 1974. - М.: Наука, 1974. - С. 7-25.

71. Уемов А.И. Системный подход и общая теория систем. М.: Мысль,1978.-271 с.

72. Урманцев Ю.А. Начала общей теории систем. В кн.: Системный анализ и научное знание. - М.: Наука, 1978.

73. Перегудов Ф.И. Системное проектирование АСУ хозяйством области. -М.: Статистика, 1977.

74. Афанасьев В.Г. Моделирование как метод исследования социальных систем. В кн.: Системные исследования. Ежегодник, 1982. - М.: Наука, 1982 - С. 2646.

75. Гвишиани Д.М. Материалистическая диалектика филилософская основа системных исследований. - В кн.: Системные исследования. Ежегодник, 1979. — М.: Наука, 1980.-С. 7-28.

76. Уемов А.И. Основы формального аппарата параметрической общей теории систем. В кн.: Системные исследования. Ежегодник, 1984. - М.: Наука, 1984. -С. 152-180.

77. Кузьмин В.П. Различные направления разработки системного подхода и их гносеологические обоснования. В кн.: Системные исследования. Ежегодник, 1984. - М.: Наука, 1984. - С. 7-31.

78. Яблонский А.И. Методологические вопросы анализа сложных систем. — В кн.: Системные исследования. Ежегодник, 1984. М.: Наука, 1984. - С. 52-65.

79. Яблонский А.И. Наука в системе глобального моделирования. В кн.: Системные исследования. Ежегодник, 1980. -М.: Наука, 1981. - С. 174-195.

80. Юдин Б.Г. Некоторые особенности развития системных исследований. -В кн.: Системные исследования. Ежегодник, 1980. М.: Наука, 1981. - С. 7-23.

81. Цыгичко В.Н. Принципы системности в теории принятия решений. В кн.: Системные исследования. Ежегодник, 1984. - М.: Наука, 1984. - С. 368-379.

82. Квейд Э. Анализ сложных систем. М.: Сов. радио, 1969. - 519 с.

83. Клиланд Д., Кинг В. Системный анализ и целевое управление. М.: Сов. радио, 1974.

84. Янг С. Системное управление организацией. М.: Сов. радио, 1972.

85. Черчмен У., Акофф Р., Арнофф Л. Введение в исследование операций /

86. Пер. с англ. М.: Наука, 1968.

87. Дитрих Я. Проектирование и конструирование: системный подход / Пер. с польск. -М.: Мир, 1981. -456 с.

88. Блауберг И.В., Мирский Э.М., Садовский В.Н. Системный подход и системный анализ. В кн.: Системные исследования. Ежегодник, 1982. - М.: Наука, 1982. -С. 47-64.

89. Смирнов Г.А. Основы формальной теории целостности. В кн.: Системные исследования. Ежегодник, 1980. - М.: Наука, 1981. - С. 255-283.

90. Дубова Н. Интегрированные системы управления распределенной корпорацией // Открытые системы. 1998. - № 1.

91. Дворникова Е. Корпорационная информационная система «Эталон» // Открытые системы. 1998. - № 2.

92. Царегородцев А.В. Теоретические основы моделирования криптографических преобразований: Учебное пособие. М.: Изд-во РУДН, 2003. - 48 с.

93. Пятибратов А.П., Гудыно Л.П., Кириченко А.А. Вычислительные системы, сети и телекоммуникации. М.: Финансы и статистика, 1998.

94. Ломакин И.В., Никольский С.Н., Николаев В.Ф. Некоторые результаты по разработке процессов функционирования целеустремленных систем // Теория систем и разработка АСУ: Тез. док. 2-й Московской научной конференции. — Москва, 1976. С. 4.

95. Царегородцев А.В. Автоматизированная разработка платформ безопасности распределенных информационно-управляющих систем: Учебное пособие. М.: Изд-во РУДН, 2002. - 48 с.

96. Мартынов В.И. Синтез первичных сетей связи из неустойчивых элементов // Автоматика и телемеханика. М., 1998. - № 9. - С. 36-52.

97. Царегородцев А.В. Разработка метода распараллеливания каналов для сохранения целостности информации в распределенных информационно-управляющих системах // Вестник РУДН, Сер. Инженерные исследования. М., 2001.-№ 1.-С. 22-33.

98. Павлов В., Сафаров М. PRODIS управление производственными ресурсами // Открытые системы. - 1998. - № 2.

99. Царегородцев А.В. Теоретические основы построения платформ безопасности информационно-управляющих систем. М.: Изд-во РУДН, 2003. - 112 с.

100. Царегородцев А.В. Разработка концептуальной модели метасистемы автоматизированного проектирования платформ безопасности информационно-управляющих систем // Приборы и системы. Управление, контроль, диагностика. -М., 2003.-№ 5.-С. 1-6.

101. Дубова Н., Кутукова Е. Unicenter TNG управление распределенной корпорацией // Открытые системы. -1998. - № 2.

102. Катышев С. Об одной концепции управления распределенными ресурсами // Открытые системы. 1998. - № 3.

103. Царегородцев А.В. Принципы построения защищенных распределенных информационно-управляющих систем // Приборы и системы. Управление, контроль, диагностика. М., 2004. - № 3. - С. 1-6.

104. Баранов И.А., Толмачев С.А. Алгебраическая модель архитектуры отказоустойчивой вычислительной системы // Автоматика и телемеханика. М., 1991. — № 1.-С. 175-180.

105. Полукеев О., Коваль Д. Моделирование бизнеса и архитектура информационной системы // СУБД. 1995. - № 4.

106. Бусленко Н.П. Моделирование сложных систем. М.: Наука, 1978.399 с.

107. Калман Р., Фалб П., Арбиб М.А. Очерки по математической теории систем / Пер. с англ. М.: Мир, 1971. - 400 с.

108. Арбиб М.А., Мейнс Э.Дж. Основания теории систем: разложимые системы. В кн.: Математические методы в теории систем / Под ред. Куравлева Ю.И. -М.: Мир, 1979.-С. 7-48.

109. Андерсон Б.Л., Арбиб М.А., Мейнс Э.Дж. Основания теории систем: конечные и неконечные условия. В кн.: Математические методы в теории систем / Под ред. Куравлева Ю.И. - М.: Мир, 1979. - С. 49-133.

110. Гисин В.Б., Цаленко М.111. Алгебраическая теория систем и ее приложения. В кн.: Системные исследования. Ежегодник, 1984. - М: Наука, 1984. - С. 66-82.