**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»**

Кафедра БІТ



**Реферат на тему:**

**“Огляд протоколу TLS”**

**З дисципліни**

**“Системи та засоби автентифікації”**

Львів – 2011

**Зміст**

1. Опис протоколу
2. Ієрархія організації інформаційного обміну
3. Алгоритм перетворення інформації при обміні
4. Атрибути сесії
5. Атрибути з'єднання
6. Типи повідомлень
7. Алгоритм процедури встановлення з'єднання по протоколу TLS handshake
8. Handshake в деталях
9. Алгоритми, що використовуються в TLS

**Опис протоколу**

TLS (англ. Transport Layer Security - транспортний рівень безпеки) - криптографічний протокол, що забезпечує захищену передачу даних між вузлами в мережі Інтернет.

TLS-протокол заснований на протоколі Netscape SSL версії 3.0 і складається з двох частин - TLS Record Protocol і TLS Handshake Protocol. Відмінності між SSL 3.0 і TLS 1.0 незначні.

TLS надає можливості автентифікації і безпечної передачі даних через Інтернет з використанням криптографічних засобів. Часто відбувається лише автентифікація сервера, в той час як клієнт залишається нерозпізнаним. Для взаємної автентифікації кожна зі сторін повинна підтримувати інфраструктуру відкритого ключа (PKI), яка дозволяє захистити клієнт-серверні додатки від перехоплення повідомлень, редагування існуючих повідомлень і створення підроблених.

TLS включає в себе три основних фази:

1. Діалог між сторонами, метою якого є вибір алгоритму шифрування
2. Обмін ключами на основі криптосистем з відкритим ключем або автентифікація на основі сертифікатів.
3. Передача даних, що шифруються за допомогою симетричних алгоритмів шифрування.

Автентифікація може бути односторонньою (автентифікація сервера клієнтом), взаємною (зустрічна автентифікація сервера і клієнта).

**Ієрархія організації інформаційного обміну**

Ієрархія інформаційного обміну включає в себе сесії, з'єднання і потік повідомлень в з'єднанні. Потік повідомлень при великій довжині розбивається на фрагменти з пакетною передачею фрагментів. В одній сесії може бути реалізовано кілька з'єднань, довільно рознесених за часом. У кожному з'єднанні може бути оброблений необхідний потік повідомлень.

**Алгоритм перетворення інформації при обміні**

Алгоритм перетворення інформації при обміні з використанням протоколу TLS включає операції:

1. прийом від протоколу верхнього рівня потоку даних, що не інтерпретуються в блоках довільного розміру;
2. фрагментація прийнятих з верхнього рівня даних у структуровані блоки (фрагменти) протоколу TLS. Розмір фрагмента - не більше 214 байт;
3. компресія фрагментів;
4. Хешування фрагментів (використовується ключовий MAC);
5. конкатенація фрагмента і результату його хешування (розширений фрагмент);
6. шифрування розширеного фрагмента;
7. передача зашифрованого розширеного фрагмента з доданим відкритим заголовком протоколу транспортного рівня (наприклад, TCP).

При прийомі інформації застосовується зворотна послідовність операцій.

**Атрибути сесії**

Сесія характеризується наступними атрибутами:

1. ідентифікатор сесії (випадкове число, 32 байта, задається сервером при відкритті сесії);
2. метод компресії;
3. сертифікат сервера (опціонально);
4. сертифікат клієнта (опціонально);
5. специфікація алгоритмів і параметрів захисту (алгоритми шифрування і MAC, криптографічні параметри);
6. master secret (використовується при генерації ключів шифрування, ключів MAC, векторів ініціалізації);
7. прапор, який дозволяє / забороняє нові з'єднання в сеансі.

**Атрибути з'єднання**

До атрибутів сполуки відносяться:

1. client\_random - випадкові 32 байта, що задаються клієнтом;
2. server\_random-випадкові 32 байта, що задаються сервером;
3. client write MAC secret (ключ клієнта для ключового хешування);
4. server write MAC secret (ключ сервера для ключового хешування);
5. client write key (ключ, який використовується для шифрування даних клієнтом і розшифрування їх сервером);
6. server write key (ключ, який використовується для шифрування даних сервером і розшифрування їх клієнтом);
7. client write IV, server write IV (вектори ініціалізації, використовувані клієнтом і сервером відповідно);
8. порядковий номер з'єднання (підтримується незалежно для переданих і прийнятих повідомлень).

Вектори ініціалізації використовуються в блокових шифрах. Вектор ініціалізації задається для першого фрагмента повідомлення у з'єднанні; для наступних фрагментів вектор ініціалізації формується з кінцевого блоку зашифрованого тексту попереднього фрагмента.

Порядкові номери з'єднань підтримуються незалежно для переданих і прийнятих повідомлень. При зміні сесії, зміні специфікації алгоритмів і параметрів захисту нумерація сполук починається з 0. Діапазон нумерації: 0 - 264-1.

З'єднання асоціюється з одного сесією.

**Типи повідомлень**

У протоколі TLS використовуються наступні типи повідомлень:

1. Hello message (ClientHello, ServerHello);
2. Change cipher specs message (зміна специфікації алгоритмів і параметрів захисту);
3. Key exchange message (передача ключа обміну ключами шифрування і МАС клієнта, сервера);
4. Alert message (попередження, оповіщення про фатальну помилку);
5. Application\_data message (передача даних);
6. Finished message (повідомлення про можливість роботи в створеній сесії).

**Алгоритм процедури встановлення з'єднання по протоколу TLS handshake**

Клієнт і сервер, що працюють за TLS, встановлюють з'єднання, використовуючи процедуру handshake («рукостискання»). Протягом цього handshake клієнт і сервер приймають угоду щодо параметрів, використовуваних для встановлення захищеного з'єднання.

Послідовність дій при встановленні TLS-з'єднання:

* клієнт підключається до сервера, що підтримує TLS, і запитує захищене з'єднання;
* клієнт надає список підтримуваних алгоритмів шифрування і хеш-функцій;
* сервер вибирає зі списку, наданого клієнтом, найбільш стійкі алгоритми, які також підтримуються сервером, і повідомляє про свій вибір клієнта;
* сервер відправляє клієнту цифровий сертифікат для власної автентифікації. Зазвичай цифровий сертифікат містить ім'я сервера, ім'я довіреної центру сертифікації і відкритий ключ сервера;
* клієнт може зв'язатися з сервером довіреного центру сертифікації і підтвердити автентичність переданого сертифіката до початку передачі даних;
* для того щоб згенерувати сеансовий ключ для захищеного з'єднання, клієнт шифрує випадково сгенерированную цифрову послідовність відкритим ключем сервера і посилає результат на сервер. Враховуючи специфіку алгоритму асиметричного шифрування, використовуваного для встановлення з'єднання, тільки сервер може розшифрувати отриману послідовність, використовуючи свій закритий ключ;

**Handshake в деталях**

Згідно з протоколом TLS програми обмінюються записами, інкапсулює (зберігають усередині себе) інформацію, яка повинна бути передана. Кожна із записів може бути стиснута, доповнена, зашифрована або ідентифікована MAC (код автентифікації повідомлення) залежно від поточного стану з'єднання (стану протоколу). Кожен запис в TLS містить наступні поля: content type (визначає тип вмісту записи), поле, яке вказує довжину пакету, і поле, яке вказує версію протоколу TLS.

Коли з'єднання тільки встановлюється, взаємодія йде по протоколу TLS handshake, content type якого 22.

Нижче описаний простий приклад встановлення з'єднання:

1. Клієнт посилає повідомлення ClientHello, вказуючи найбільш останню версію підтримуваного TLS протоколу, випадкове число і список підтримуваних методів шифрування і стиснення, які підходять для роботи з TLS.
2. Сервер відповідає повідомленням ServerHello, що містить: обрану сервером версію протоколу, випадкове число, послане клієнтом, відповідний алгоритм шифрування і стиснення зі списку наданого клієнтом.
3. Сервер посилає повідомлення Certificate, яке містить цифровий сертифікат сервера (залежно від алгоритму шифрування цей етап може бути пропущений)
4. Сервер може запросити сертифікат у клієнта, в такому випадку з'єднання буде взаємно автентифікований.
5. Сервер відсилає повідомлення ServerHelloDone, ідентифікує закінчення handshake.
6. Клієнт відповідає повідомленням ClientKeyExchange, яке містить відкритий ключ PreMasterSecret або нічого (знову ж таки залежить від алгоритму шифрування).
7. Клієнт і сервер, використовуючи ключ PreMasterSecret і випадково згенеровані числа, обчислюють загальний секретний ключ. Вся інша інформація про ключ буде отримана із загального секретного ключа (і згенерованих клієнтом і сервером випадкових значень).
8. Клієнт посилає повідомлення ChangeCipherSpec, яке вказує на те, що вся подальша інформація буде зашифрована встановленим в процесі handshake алгоритмом, використовуючи загальний секретний ключ. Це повідомлення рівня записів і тому має тип 20, а не 22.
9. Клієнт посилає повідомлення Finished, яке містить хеш і MAC, згенеровані на основі попередніх повідомлень handshake.
10. Сервер намагається розшифрувати Finished-повідомлення клієнта і перевірити хеш і МАС. Якщо процес розшифровки або перевірки не вдається, handshake вважається невдалим, і з'єднання повинно бути обірвано.
11. Сервер посилає ChangeCipherSpec і зашифроване повідомлення Finished, і в свою чергу клієнт теж виконує розшифровку та перевірку.

**Алгоритми, що використовуються в TLS**

У даній поточної версії протоколу доступні наступні алгоритми:

* Для обміну ключами та перевірки їх достовірності застосовуються комбінації алгоритмів: RSA (асиметричний шифр), Diffie-Hellman (безпечний обмін ключами), DSA (алгоритм цифрового підпису) і алгоритми технології Fortezza.
* Для симетричного шифрування: RC2, RC4, IDEA, DES, Triple DES або AES;
* Для хеш-функцій: MD5 або SHA.