Einführung in die Informationssicherheit – Bericht Team 7 - P4.1 Attacken auf TLS

Was habt ihr gemacht?

Wir haben uns die Schwächen der verschiedenen Security-Algorithmen angesehen und haben die größte Schwäche bei SSL 2.0 erkannt, wo ein Man-in-the-middle die Pakete des Clients an den Server sendet, direkt manipuliert werden können. Dem Server kann so vortäuschen werden, dass der Client eine nur eine alte Version mit schwachem Cipher-Suite beherrscht.

Was kann eure Implementierung?

Unsere Implementierung behebt die Fehler des vorgegebenen Proxies beim parsen der URL und dem weiterleiten der Pakete an den richtigen Port. Des weiteren überprüfen wir die Paketstruktur, und geben eine Message aus, wenn wir ein ClientHello Paket erkennen.
Das HelloClient Paket wird zu Beginn des Verbindungsaufbaus vom Client zum Server geschickt.

Es beinhaltet im Byte mit Offset 0 eine 22, welches es als Handshake-Paket identifiziert. Auf Offset 1 und 2 wird dann die Major und Minor version gespeichert. Major 3 und Minor 0 bedeutet dabei SSL 3.0, mit Minor 1 bedeutet es TLS 1.0, und so weiter. Der Client sollte den höchsten von ihm beherrschten Standard angeben. Daher wenn dieser Teil nur SSL2 ist, kann die Nachricht direkt manipuliert werden.
Auf den folgenden 2 Bytes wird dann die Gesamtlänge der Nachricht gespeichert.
Im Anschluss werden dann ein oder mehrere Message-Blöcke gespeichert. Diese bestehen aus einem Byte für den Typ der Nachricht, bei einem ClientHello ist dieser Wert 1. Dann folgt auf den nächsten 3 Bytes die Länge des Blockes von Daten für diese Handshake-Nachricht.
In den Daten des Hello Client Pakets kann eine Session-ID vorhanden sein. Mit dieser kann eine Session mit den ausgehandelten Sicherheitsparametern wieder aufgenommen werden.
Wichtiger ist jedoch die Liste von Cipher Suites die der Client unterstützt, sortiert nach dessen Präferenzen, welche mitgesendet wird. Der Server wählt dann eine der Möglichkeiten aus, oder sendet einen Handshake Failure Alert und schließt die Verbindung.
Wir erkennen das Hello Client paket also anhand der 22 im ersten, und der 1 im 5ten Byte.

Was sind die Resultate eurer Implementierung?

Während TLS 1.2 (noch?) sehr sicher ist, gibt es grobe Schwächen bei SSL2 und auch SSL3 ist bei weitem nicht mehr sicher durch das fälschen von Zertifikaten mit Wildcards.

Probleme während der Implementierung:

Nach der Implementierung für das Erkennen der ClientHello Nachricht hatten wir einige Probleme ein Downgrade durchzuführen. Ab SSL 3.0 ist keine Modifikation des Handshakes mehr möglich ist und ein **Handshake Failure** wird geliefert. In der E-Mail die wir erhielten, wurden uns 2 Möglichkeiten für die weitere Vorgehensweise vorgeschlagen. Wir entschieden uns schließlich dafür die Erkennung einer solchen Modifikation zu erklären.

Möglichkeiten zur Erkennung einer Downgrade Attacke:

Seit der Version 3.0 verwendet SSL eine zusätzliche Nachricht im Handshake um eine Manipulation des Handshakes (Man-in-the-middle attack) zu erkennen. Es ist nun nicht mehr möglich die Liste der unterstützten Cipher-Suites soweit zu manipulieren, dass sie nur mehr schwache Cipher-Suites enthält.

Die **Finished** Nachricht wird nach einer **ChangeCipherSpec** Nachricht gesendet. Die **ChangeCipherSpec** ist keine Handshake Nachricht, sondern besitzt einen eigenen Content-Type und markiert den Punkt, aber der der Client bzw. der Server zu der ausgewählten Cipher-Suite wechseln. Jegliche darauffolgenden Nachrichten sind nun mit dieser Cipher-Suite verschlüsselt.

Der Client sendet zuerst eine **ChangeCipherSpec** Nachricht, gefolgt von einer **Finished** Nachricht. Die **Finished** Nachricht ist eine verschlüsselte Checksumme über alle bisherigen gesendeten Handshake Nachrichten. Für die Berechnung der Checksumme werden sowohl Client als auch Server Nachrichten verwendet. Da diese Nachricht nach **ChangeCipherSpec** gesendet wird, verwendet sie schon die ausgewählte Verschlüsselung. Der Server erhält nun die **Finished** Nachricht, überprüft deren Checksumme und erhält somit eine Bestätigung, dass die Kommunikation nicht manipuliert wurde. Es ist nicht möglich, dass die **Finished** Nachricht trotz einer Manipulation korrekt ist.

Nun sendet der Server ebenfalls eine **ChangeCipherSpec** Nachricht, gefolgt von einer **Finished** Nachricht die nun vom Client ausgewertet wird und die Handshake Phase ist hiermit erfolgreich beendet falls beide Checksummen korrekt sind.

Sollte eine ein MITM jedoch nicht durch eine Handshake Manipulation eine Downgradeversuch starten, sondern durch Verbindungsabbrüche, so dass es den Anschein hat als ob der Server die angefragte TLS Version nicht unterstützt, so gibt es eine andere Methode um dies zu erkennen.

Dies wird durch den zusätzlichen Cipher-Suite Wert **TLS_FALLBACK_SCSV** (signaling cipher suite value) erreicht. Muss ein Client einen TLS Handshake mit einer niedrigeren Version initiieren, so sendet er nun **TLS_FALLBACK_SCSV** als Cipher-Suite mit. Erkennt der Server nun **TLS_FALLBACK_SCSV** in der Cipher-Suite Liste und die höchste vom Server unterstützte TLS Version ist höher als die vom Client angegebenen Version so sendet der Server eine **inappropriate_fallback** Nachricht an den Client. Sollte der Server **TLS_FALLBACK_SCSV** jedoch nicht erkennen so wird mit dem Handshake normal fortgefahren.