Hashes, Blockcipher
Projekt u23

Jan Lühr

October 18, 2015

1 Blockcipher

Programmiere den Electronic-Codebook- (ECB), Ciber-Block-Chaining- (CBC) und den Counter-Modus (CTR) selbst. Wähle eine Variante:

Einfach Verschlüssele das Wort AAAABBBBAAAA in Sage.
Cipher: XOR, Blockgröße 1 Byte
Wie unterscheidet sich das Wort zwischen den Modi?

Schwieriger Verschlüssele ein Bitmap-Bild (.bmp) mit Python.
Cipher: AES-128 (oder DES).
Wie unterscheidet sich das Bild zwischen den Modi?

Hinweis: .bmp-Header nicht verschlüsseln oder nachträglich wieder herstellen.

2 Hashing

2.1 SHA-3 / Keccak

Installiere Keccak. Welche Prüfsummen hat diese PDF-Datei?

2.2 MD5 - Eigenschaften Angriffe

Informiere Dich über die Funktionsweise von MD5 und die verschiedenen Angriffe.

1. Welche Probleme und Angriffe sind bekannt?
2. Wie konnten Forscher ein mit MD5 signiertes TLS-Zertifikat fälschen?
3. Was sind Rainbow-Tabellen?

2.3 Sicheres und Unsicheres Hashing

Sind die folgenden Verfahren sicher? Welche Probleme siehst Du?

2.3.1 Signierte Downloads

Alice bietet ein Programm im Internet zum Download an. Damit es beim Transport nicht manipuliert wird, berechnet sie eine SHA-3 Summe des Programms. Die Prüfsumme signiert Sie mit ihrem GPG-Key. Programm, SHA-3-Summe und Signatur stellt sie auf ihrem Webserver zur Verfügung.
Kann Bob sicher überprüfen, ob er das Programm korrekt heruntergeladen hat?

2.3.2 Welches Kennwort passt?

Adrians System speichert von jedem Kennwort einen Hash. Um ein Kennwort zu überprüfen, hashed er es und vergleicht die beiden Hashes. So muss er die Kennwörter nicht im Klartext speichern. Als Backup speichert er die Hashes in der Cloud.

- Adrian verwendet MD5. Mallory kann den Hash `efce5bb6c4a73bf05bc66ab889712230` aus der Cloud laden. Wie ist das Kennwort?
- Adrian verwendet SHA-3. Ist das Verfahren sicherer? Was sollte Adrian zusätzlich machen?

2.3.3 HMAC Firewalling

Anton hat einen VPN-Server. Die Überprüfung einer neuen, eingehenden Verbindung (Benutzername, Public-Key) benötigt viel Zeit und Ressourcen — viele neue Verbindungen überlasten den Server schnell.
Mallory nutzt es aus. Er überlastet den Server, in dem er viele Anfragen mit falschen Zugangsdaten sendet.

Um seinen Server zu schützen denkt sich Anton ein schnelles Verfahren aus:

1. Jeder VPN-Teilnehmer hat zusätzlich zum Public / Private Key das gleiche, einfache Kennwort - z.B. `hjsao9eks`
2. Wird eine neue Verbindung aufgebaut, so übermittelt der Teilnehmer einen MD5-MAC mit Kennwort, IP-Adressen und eine Nonce. Beispiel:

$$MD5(\texttt{'hjsao9eks'}, \texttt{'192.168.0.2'}, \texttt{'192.168.0.1'}, \texttt{0xF50767A0})$$

3. Baut ein Teilnehmer eine Verbindung zum VPN-Server auf, so überprüft Anton die MAC. Tritt ein Fehler auf, so wird die Verbindung verworfen.

Ist das Verfahren sicher? Was müsste Mallory tun, um den Server weiterhin zu überlasten? Helfen ihm Kollisionen in MD5 und Rainbow-Tabellen?