Übungsblatt 10: Netzwerkkommunikation

Vorstellung in den Tutorien am 17. - 19. Januar 2023

10.1 Lehrevaluation

Auch in diesem Semester führt die TU Berlin wieder eine Lehrevaluation durch. Im Rahmen dieser Evaluation haben Sie die Möglichkeit, **anonym** die Qualität der von Ihnen besuchten Veranstaltungen zu bewerten und uns Feedback zukommen zu lassen.

Die Lehrevaluation erfolgt über Online-Fragebögen, wobei die Vorlesung und Übung jeweils in einem Separaten Fragebogen bewertet werden. Wir bitten Sie, beide Fragebögen ehrlich und möglichst vollständig auszufüllen.

Die Teilnahme an der Evaluation ist natürlich nicht verpflichtend, wir würden uns aber über möglichst zahlreiches und ehrliches Feedback freuen!

Fragebogen für die **Vorlesung**:
`https://befragung.tu-berlin.de/evasys/online.php?p=TFSMC`

Fragebogen für die **Übung**:
`https://befragung.tu-berlin.de/evasys/online.php?p=X4ZZL`

10.2 Der TCP/IP-Protokollstack

Machen Sie sich mit dem TCP/IP-Protokollstack vertraut und finden Sie für jede Ebene mindestens zwei Protokolle.

———Beginn Lösung———

1. **Application Layer:** HTTP(S), FTP, SMTP, IMAP
2. **Transport Layer:** TCP, UDP
3. **Internet Layer:** IP, ICMP, IGMP
4. **Link Layer:** Ethernet, ARP, DSL, ISDN

———Ende Lösung———

Musterlösung

10.3 Sicherheitsfunktionen

Nennen und beschreiben Sie die vier wichtigsten Sicherheitsfunktionen, die für eine sichere Kommunikation im Web notwendig sind.

———Beginn Lösung———

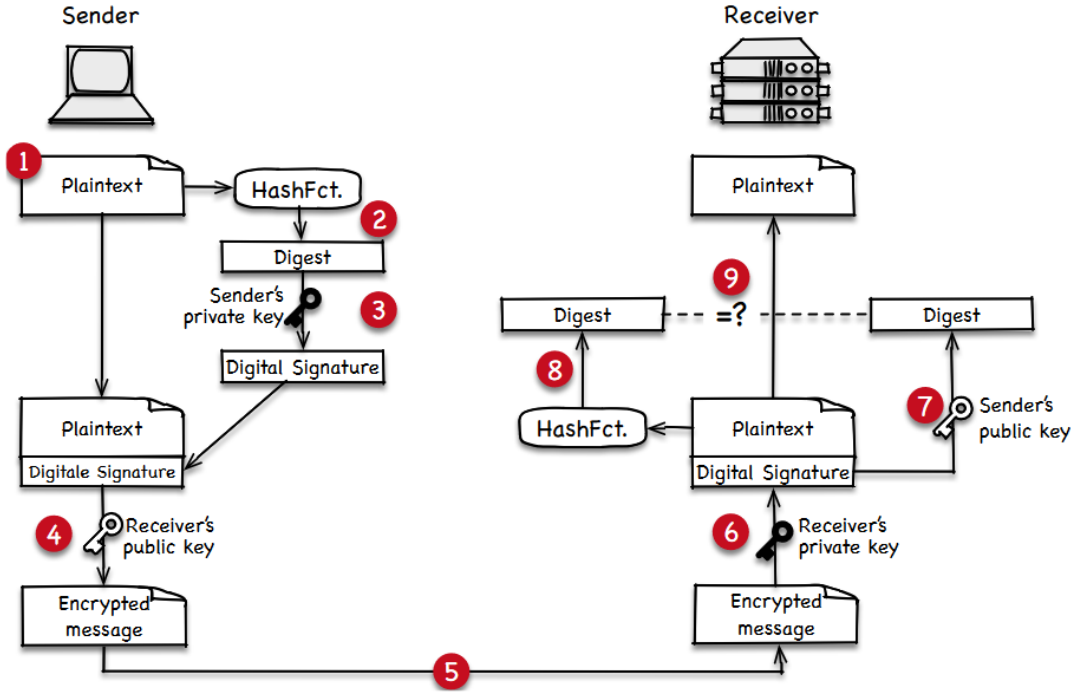
1. **Authentisierung:** Nachweis einer Person oder Entität, dass sie tatsächlich ist, wer sie vorgibt zu sein
2. **Authentifizierung:** Überprüfung der behaupteten Authentisierung durch die Person oder Entität gegenüber der die Authentisierung erfolg ist
3. **Vertraulichkeit:** Schutz vor unbefugter Preisgabe von Informationen
4. **Integrität:** Daten können nicht unbemerkt, d.h. durch Angreifer, verändert werden

———Ende Lösung———

10.4 Ablauf einer sicheren Kommunikation

In der Vorlesung haben Sie den Ablauf einer asymmetrisch verschlüsselten Kommunikation sowie einer digitalen Signatur kennengelernt. Fassen Sie kurz zusammen, wofür asymmetrische Verschlüsselung und digitale Signaturen dienen.

Die untenstehende Abbildung fasst beide Abläufe in einem Diagramm zusammen. Beschriften und erklären Sie die markierten Schritte! Achten Sie dabei besonders darauf, welcher Schlüssel im jeweiligen Schritt verwendet wird.



———Beginn Lösung———

1. Sender erstellt Nachricht mit Inhalt
2. Mathematische Berechnung, genannt *Hashfunktion* wird auf Nachricht angewendet, um einen *Digest* zu generieren
3. Sender benutz **seinen privaten Schlüssel**, um den Digest zu verschlüsseln, woraus sich die *digitale Signatur* ergibt - niemand kann die digitale Signatur des Senders replizieren
4. Sender verschlüsselt sowohl die originale Nachricht als auch die digitale Signatur mit dem **öffentlichen Schlüssel des Empfängers**
5. Das verschlüsselte Bündel aus Nachricht und digitaler Signatur wird an den Empfänger übertragen
6. Der Empfänger nutzt **seinen privaten Schlüssel**, um das empfangene Bündel zu entschlüsseln
7. Der Empfänger nutzt den **öffentlichen Schlüssel des Senders**, um die digitale Signatur zu entschlüsseln und den Digest der Originalnachricht zu extrahieren
8. Mithilfe der selben *Hashfunktion* aus Schritt 2 erstellt der Empfänger einen *Digest* der empfangenen und in Schritt 6 entschlüsselten Nachricht
9. Der Empfänger vergleicht diesen Digest mit dem der Originalnachricht, um die Authentität der empfangenen Nachricht zu verifizieren

———Ende Lösung———

Musterlösung

10.5 HTTP: Methoden

In der Vorgabe finden Sie eine Datei namens *server.js*. Hierbei handelt es sich um eine kleine *Node.js* Server-Anwendung. Führen Sie diese mit dem Konsolenbefehl `node server.js` aus. Dies startet einen lokalen HTTP-Server auf Port 8080, den Sie über die URL `http://localhost:8080` erreichen können. Auf dem Server liegt eine kleine Filmdatenbank, deren Einträge folgendes Format haben:

```
id: Number, title: String, release: Number
```

Einträge können durch Angabe ihrer ID abgerufen werden. Nutzen Sie im Folgenden *Postman*, um Anfragen an diesen HTTP-Server zu schicken:

1. Lassen Sie sich durch eine OPTIONS-Anfrage ausgeben, welche HTTP-Methoden vom Server unterstützt werden.
2. Führen Sie eine GET-Anfrage aus, um sich den Eintrag mit der `id=0` ausgeben zu lassen. Womit antwortet der Server, wenn Sie keine ID als Parameter angeben?
3. Fügen Sie mithilfe von POST-Anfragen weitere Einträge in die Filmdatenbank ein. Geben Sie dazu `title` und `release` der jeweiligen Filme im `x-www-form-urlencoded` Format an.

4. Verändern Sie einen existierenden Eintrag durch eine PUT-Anfrage. Geben Sie dazu die `id` sowie den neuen `title` und `release` des Eintrags, den Sie verändern möchten, im `x-www-urlencodedFormat` an.

5. Löschen Sie einen Eintrag Ihrer Wahl, indem Sie eine DELETE-Anfrage mit der ID des Eintrags als Parameter ausführen.

Musterlösung