Universität Heidelberg
Sommersemester 2019

Lehrstuhl für Parallele und Verteilte Systeme (PVS)
Diego Costa, Artur Andrzejak

10. Übung zur Vorlesung „Betriebssysteme und Netzwerke“ (IBN)

Abgabedatum: 02.07.2019, 11:00 Uhr

Aufgabe 1 (2 Punkte)

Auf Seiten wie http://www.speedmeter.de können Sie Leistungsmerkmale Ihrer Internetanbindung messen.

a) Welche der Größen aus Vorlesung N01 werden gemessen? Welche sind wichtig für welche Anwendungen?

b) Hängt das Ergebnis der Tests davon ab, wo Sie sich im Internet im Verhältnis zum Server der Testseite befinden (also „nah” bzw. „weit” entfernt)? Erklären Sie, warum das so ist. Was könnte die Messungen noch beeinflussen?

Aufgabe 2 (2 Punkte)

In der Vorlesung wurden die nachfolgenden Schichten besprochen, aus denen sich der Protokollstapel zusammensetzt (Bitübertragungsschicht wurde hier ausgenommen). Beachten Sie, dass die Antworten für die Anwendungsschicht bei dieser Aufgabe von der Anwendung abhängt. Nennen Sie für die Anwendungsschicht zwei verschiedene Beispiele.

- Anwendungsschicht
- Transportschicht
- Netzwerkschicht
- Sicherungsschicht

a. Wie nennt man jeweils die gekapselten Nachrichten, nachdem jeweils der Header der Schicht hinzugefügt wurde?

b. Welche Endpunkte werden in welcher Schicht jeweils verbunden?

c. Geben Sie für jede der obigen Schichten an, wodurch das Ziel eines Pakets eindeutig adressiert werden kann.

Aufgabe 3 (2 Punkte)

Man kann Emails über Mail-Clients (wie Mozilla Thunderbird) versenden, oder mit einem Web-Client (Browser) über Web-Interfaces, wie sie viele Anbieter bereitstellen. Erklären Sie den Unterschied im Sinne der Anwendungsprotokolle, die benutzt werden. Geben Sie einen Vorteil beider Praktiken gegenüber der jeweils anderen für den User an.

Aufgabe 4 (1 Punkt)

Betrachten Sie die vier Anteile der Gesamtverzögerung an jedem Übertragungsknoten, die auf der Folie *Gesamtverzögerung (pro Übetragungsknoten)* der Vorlesung N01 dargestellt sind. Angenommen, Sie sind der CTO eines Tier-1-ISPs und wollen die Verzögerung in Ihrem Netzwerk reduzieren. Bei welchen dieser Verzögerungsarten können Sie effektive Verbesserungen erreichen, und wie?

Aufgabe 5 (3 Punkte)

Erinnern Sie sich an das Konzept der Kapselung aus Vorlesung N02. Nehmen Sie an, Sie betrachten ein Protokoll P aus einer Schicht S paketvermittelter Kommunikation. Es biete verschiedene optionale Dienste an eine höhere Schicht an, unter anderem ZIP-Komprimierung des Payloads. Bei jeder Anwendung eines Dienstes werde der Payload einmal zusätzlich durch P verkapselt.

Es ist möglich, ZIP-Daten zu fingieren, die bei der Dekompression dieselben Daten reproduzieren[1]. Nehmen Sie an, ein Hacker schreibt eine Software, die ein Paket erzeugt, das ein solches rekursives Paket als komprimiertes Payload enthält. Beim Entpacken des Payloads ergibt sich erneut dasselbe Paket. Der empfangende Teilnehmer des Protokolls wiederholt den Entpackungsprozess unbegrenzt lange. Damit wird der Prozess oder der ganze Rechner zum Absturz gebracht oder zumindest der sonstige Datenverkehr beeinträchtigt.

a) Beantworten Sie folgende Fragen für jede der Netzwerkschichten, aus der das Protokoll P stammen kann (Transport-, Netzwerk- oder Sicherungsschicht): Welcher Host im Netzwerk wird durch diesen Angriff gestört oder sogar zum Absturz gebracht? Von welchem Host aus muss der Angreifer sein Paket verschicken, wenn er ein bestimmtes Ziel im Sinn hat?

b) Welche Sicherheitsmechanismen könnten im Protokoll P festgeschrieben werden, um solchen Missbrauch zu verhindern? Welcher Teilnehmer des Protokolls müsste seine Implementation ändern?

Aufgabe 6 (4 Punkte)

In dieser Aufgabe sollen Sie das HTTP-Protokoll mit Hilfe des Programmes Wireshark[2] genauer kennenlernen. Installieren Sie dazu Wireshark auf Ihrem Rechner. Eine Einführung finden Sie unter

[1] http://research.swtch.com/zip
[2] http://www.wireshark.org/

„Getting Started" auf den Webseiten[3] zum Buch von Kurose et al.. Lesen Sie die Aufgabenstellung zum Wireshark Lab „HTTP"[4] durch.

a) Führen Sie die Experimente in dem Abschnitt 1 („The Basic HTTP GET/response interaction") durch, und beantworten Sie Fragen 4, 5 und 6.

b) Machen Sie sich mit der Funktionsweise der bedingten GET-Methode vertraut (Abschnitt 2.2.6 im Buch von Kurose et al. oder etwa hier[5]). Arbeiten Sie den Abschnitt 2 („The HTTP CONDITIONAL GET/response interaction") durch, und beantworten Sie (kurz) Fragen 8 bis 11.

c) Arbeiten Sie den Abschnitt 3 („Retrieving Long Documents") durch. Schalten Sie (falls vorhanden) in Wireshark die Option für TCP Reassembly aus (siehe hier[6]). Beantworten Sie Frage 15 und geben Sie die Anzahl der Payload-Bytes in jedem der TCP-Pakete der HTTP-Response an.

d) Beantworten Sie in Abschnitt 4 („HTML Documents with Embedded Objects") die Fragen 16 und 17.

Aufgabe 7 (2 Punkte)

Lesen Sie den Artikel *Cneonction* (sic!)*: closed HTTP header*[7] und beantworten Sie folgende Fragen:

- Welches Feature wird nach der Beschreibung implementiert und zu welcher Protokollschicht wird es hinzugefügt?
- In welchen Netzwerkknoten wird die Funktion implementiert? Wie wird sie implementiert?

Aufgabe 8 (Bonus, 2 Punkte)

Ursprünglich ist HTTP ein zustandsloses Protokoll. Das heißt, über einen Request hinaus „vergisst" der Server alles über den Client dieser Sitzung. Im Jahr 1997 wurde in HTTP das Konzept des „Cookies"[8] eingeführt, mit dem der Server den Client veranlassen kann, auf dem Client-Host eine kleine Datei anzulegen, die er bei der nächsten HTTP-Sitzung mit diesem Client auslesen kann. Diese werden zum Beispiel von manchen Webservern dazu genutzt, eine „Session ID" abzuspeichern, sobald ein Nutzer sich bei der Webseite mit seinem Passwort authentifiziert hat. Dann kann er sich zum Aufrufen weiterer geschützter Seiten mit dieser ID authentifizieren. Diese IDs werden manchmal auch in der URL übergeben, statt in einem Cookie. Erklären Sie den Angriff „Session fixation" [9] und geben Sie eine Gegenmaßnahme an.

[3] http://gaia.cs.umass.edu/wireshark-labs/
[4] gaia.cs.umass.edu/wireshark-labs/Wireshark_HTTP_v7.0.pdf
[5] http://rutura jv.wordpress.com/2005/12/27/conditional-get-request/
[6] http://wiki.wireshark.org/TCP_Reassembly
[7] http://blog.eitanadler.com/2012/10/cneonction-closed-http-header.html
[8] http://tools.ietf.org/html/rfc2109
[9] https://en.wikipedia.org/wiki/Session_fixation