Sicherheit im WorldWideWeb

Thomas Osterried

IN-BERLIN e.V.

Vortragsreihe im Rahmen des

moabit.kiezlan.de

am 21.05.2011

EFRE Berlin Bezirksamt Mitte von Berlin QM Moabit-Ost

Überblick

- Sicherheit im WorldWideWeb - Zwei Teile: Theorie und Praxis
  - Was ist https?
  - Was bringt das meinem Besucher?
  - Wozu kann ich es brauchen (s.a. unten: webdav)?
  - Wo bekomme ich Zertifikate her? Wie funktioniert das System der Vertrauenshierarchie?
  - Upload / Pflege meiner Webseiten auf dem Webserver
  - Unterschiedliche Verfahren zur sicheren Authentifizierung und Übertragen der Daten am Beispiel der Zugangswege webdav, ssh/scp und ftp

Einführung

- Rückblende zur vorhergehenden Veranstaltung
  - Webseiten liegen auf Webservern
  - Protokoll: http (HyperText TransferProtocol)
    - Entwicklung seit 1991
    - Definition Protokoll:
      - Wir verwenden TCP/IP. Webserver hören üblicherweise auf einer wohldefinierten Portnummer: 80
      - Informationen werden im Klartext übertragen
      - Das Protokoll besteht im Wesentlichen drei wichtige Befehlen:
        - **PUT, POST** und **GET**

Alles Klartext

- Informationen lassen sich mitlesen
- Bei einer Webseite auf der man seinen Urlaub beschreibt eigentlich kein Problem.
  Wohl aber
  - beim Online-Banking
  - beim Lesen von E-Mail über Webmail
    - Webmail: Webserver stellt E-Mail-Postfach bereit, so daß man es im Browser lesen kann
  - weitere Ideen?

Angriffsziele..

- Verschiedene Aspekte
  - Privacy („Vertraulichkeit“)
    - z.B. in offenen WLANs oder an der „Großen Firewall“ in China können Angreifer mitlesen wer sich für was interessiert
      - in manchen Ländern bedeutet das Repression
    - *gäbe* es unverschlüsseltes Onlinebanking, würde ein Angreifer z.B. den Kontostand mitlesen können – eine „wertvolle“ Information

..Angriffsziele..

- Anmeldedaten (..was ich sende..)
  - Zugangs-Passworte
    - bei Webmail
      - der Angreifer kann dann z.B. „in meinem Namen“ unvorteilhafte E-Mail verschicken
    - internes Firmenportal
      - Zugriff auf schützenswerte Daten
  - Integrität der Daten (..was ich sende oder empfange..)
    - wurde die Information auf dem Transportweg verfälscht? (sog. Man-in-The-Middle Attacke)

..Angriffsziele

- steht auf der Webseite das was ich lese?
- empfängt der Onlineshop wirklich das was ich bestellen möchte?

- Verfügbarkeit
  - die Webseite scheint nicht erreichbar

→ gezieltes Abschneiden von Informationen

→ Prinzipien des Datenschutzes

s. Webseiten des Landesdatenschutzbeauftragen

- http://www.datenschutz-berlin.de/content/technik/begriffsbestimmungen/verfuegbarkeit-integritaet-vertraulichkeit-authentizitaet

Lösung: https..

- „Zwischenschicht“ wird eingeschoben
  - Zwischenschicht heißt SSL
    - SecureSocketLayer (Protokoll)
  - http:

1. Aufbau einer Verbindung zum Webserver
   (IP.TCP Port 80)
2. GET http://moabit.kiezlan.de/ HTTP/1.0
   („Protokoll“)
3. Transfer der angeforderten Daten vom Webserver zum Nutzer
4. Abbau der TCP-Verbindung zum Webserver

..Lösung https..

- https:
  1. Aufbau einer Verbindung zum Webserver

     (IP.TCP Port 443)
  2. Beginn des SSL-Protokolls
  3. Server präsentiert „Unterschrift“

     dazu später mehr (X.509 Zertifikate)
  4. Nutzer erkennt: das ist der der er vorgibt zu sein

     Angreifer hat diese Unterschrift nicht; der Webbrowser würde eine Warnung ausgeben
  5. Aushandlung einer Verschlüsselung

     Daten werden nicht mehr im Klartext übertragen

..Lösung: https

5. Beginn des Datentransfers innerhalb des SSL-Protokolls
6. GET https://moabit.kiezlan.de/ HTTP/1.0

   („Protokoll“)

   Angreifer:

   - sieht nicht was abgerufen werden soll
   - kann Daten nicht verfälschen
7. Transfer der angeforderten Daten vom Webserver zum Nutzer

   Angreifer:

   - sieht nicht was abgerufen wurde
   - kann Daten nicht verfälschen
8. Abbau der TCP-Verbindung zum Webserver

Worauf muß ich achten?

- URL
  - beginnt **ausnahmslos** mit https://
- Warnhinweise sollte es nicht geben
  - → **nicht einfach wegklicken!**
- Webbrowser zeigt „geschlossenes“ Schloß
- Nicht jeder Webserver bietet https
  - weil Verschlüsselung Rechenleistung zieht
  - weil Zertifikate, die die Echtheit bestätigen, Geld kosten

RootCA

- Niemand kann alle Webserver der Welt kennen
  - Deshalb gibt es sog. RootCAs
    - Root (Wurzel) ist der **Beginn** einer Zertifikatskette
      - Webbrowser kennen die RootCAs der Welt
  - Der Webserver
    - unterschreibt sozusagen auf einem von seiner RootCA (die ihm das Zertifikat ausgestellt hat) unterschriebenen Briefpapier
    - liefert sein „Server-Zertifikat“ zusammen mit dem der RootCA aus (S. Folie 9, Punkt 3, „Server präsentiert „Unterschrift“)

      → Zertifikatskette. Am Beispiel zeigen..

Server-Zertifikat..

- Das Server-Zertifikat
  - wurde von einer bekannten RootCA geprüft und unterschrieben
  - enthält die Domain (bzw. den vollständigen DNS Namen; Feld „CN=“)
    - https://www.taz.de/

      → Angesurfter Name und im Zertifikat hinterlegter müssen übereinstimmen, sonst Fehlermeldung.

      - Praxisbeispiel https://www.taz.de/ versus https://taz.de/

..Server-Zertifikat..

- kann nach entspr. Nachweis bei der RootCA den Inhaber ausweisen
  - Anderes Nachweisverfahren
    - einfache Zertifikate bestätigen Domain (CN):
      - E-Mail-Adresse wird geprüft. Gefahr: E-Mail-Postfach gehackt
    - höherer Sicherheit bieten Zertifikate, die den Firmensitz im Zertifikat unterschreiben
      - Firma legitimiert sich bei Beantragung durch Handelsregisterauszug
  - Browser macht dies besonders kenntlich
    - Browser Mozilla Firefox: grünes Feld links neben dem URL-Feld

..Server-Zertifikat..

→ höherer Vertrauensgrad

Dies machen z.B. Banken.

- Vergleiche https://www.taz.de/
  mit https://www.postbank.de/

- Wichtig:

  alle Felder im Zertifikat (Name, Aussteller, Seriennummer, Ausstelldatum und Gültigkeitsdauer, CN, Mailadrese, usw.) gehen in die Signatur und Prüfsumme mit ein und sind im Nachhinein unveränderlich.

  [Sicher - so lange, wie die Hash-Algorithmen keine Schwäche aufweisen
  → Zeitliche Sicherheit]

..Server-Zertifikat..

- ist nur eine begrenzte Zeit gültig
  - Sicherheit ist eine zeitliche Frage
    - Webserver gehackt?
      → Zertifikat abhanden gekommen?
    - Firma betrügerisch? Oder die RootCA selbst?..
    - Verschlüsselungs- oder Hashing-Alogrithmus mathematisch „gebrochen“?
- kann widerrufen werden
  - z.B. wenn Zertifikat nicht mehr vertrauenswürdig (s. Beispiel Server gehackt)

  → „CRL“ (Certificate Revocation List)

..Server-Zertifikat

- Stärkegrad und Verfahren der Unterschrift
  - Beispiel sha1 mit 2048 bit
  - macht Unterschrift „länger“ und benötigt mehr Rechenzeit
- Verschlüsselungsalgorithmus
  - z.B. RSA
- Prüfsumme „Fingerprint“ (MD5/SHA1)

  Wichtig: ist mir der richtige Fingerprint bekannt, kann ich das direkt prüfen, ohne mich auf die Zertifikatskette zu verlassen.

  - Beispiel: Beim IN-Berlin hängt der Fingerprint in Papierform mit Stempel im Schaufenster.

Absolute Sicherheit?..

- Absolute Sicherheit gibt es nicht
  - RootCA hat wohldefinierte Prozesse, damit sie überhaupt in Browser oder Betriebssystem aufgenommen werden.
    - Fehler bei der Verifikation möglich
      - Bedenke: RootCA ist „business“. Zeit ist Geld.
    - Rhetorische Frage: kann man jeder der hunderten CAs die der Browser kennt (und aus allen möglichen Ländern der Welt stammen) wirklich vertrauen?
    - Kam der CA der geheime Schlüssel abhanden?

..Absolute Sicherheit?..

- CRL / OCSP Server gerade nicht erreichbar?
- Kritik am Root-CA Prinzip
  - eine falsch spielende CA bricht die Sicherheit der Welt
    - Man stelle sich vor: RootCA unterschreibt *.com
  - Aber auch die Aufnahme von Root-CA's von Projekten wie CAcert oder DFN-CA können die Sicherheit weiter aushöhlen.
- Wie gut ist die Mathematik wirklich?
- Browser-“bugs“, die es ermöglichen Inhalte von fremden Seiten nachzuladen

..Absolute Sicherheit?
--- GIBT ES NICHT! ---

- SSL kein Garant für
  - die Qualität z.B. des besuchten Onlineshops
  - davor, daß der Webserver Schadcode (Viren) ausliefert
- Was heute sicher ist und gespeichert wird, kann schon in 10 Jahren mit minimalem Aufwand knackbar sein.

Der eigene PC

- Ist der eigene PC von Viren oder Trojanern befallen, dann nutzt auch die Sicherheit des Transportwegs nichts!

Praxisteil Zertifikate..

- Beispiele der vorherigen Seiten anschauen

  Ferner

  - taz.de
    - https://www.taz.de/

      bringt eine Warnmeldung, weil Teile der Webseite (Banner, o.ä., per http:// referenziert sind)
    - https://taz.de/
      - bringt eine Warnmeldung, weil das SSL-Zertifikat auf den CN *.taz.de (also *www*.taz.de oder *beliebiges-anderes*.taz.de ausgestellt ist). Für https://taz.de/ selbst ist das Zertifikat nicht gültig!

..Praxisteil Zertifikate..

- tatz.de

  falsche Domain („Tippfehler“)

  Davor schützt auch ein Zertifikat nicht!

  Diese Webseite hat ein „self-Signed“ Zertifikat.

- tagesschau.de ist peinlich
  - https://tagesschau.de/ geht nicht
  - https://www.tagesschau.de/ liefert Zertifikat des Hosters akamai.net (also falsches Zertifikat), und danach eine Fehlermeldung

..Praxisteil Zertifikate

→ Im Ausland sind Meldungen eines der wichtigsten deutschen öffentlich-rechtlichen Medien

- nicht verifizierbar
- verfälschbar

- gibt’s nur unverschlüsselt(!):
  - www.ard.de und www.zdf.de

- Webseiten lenken auf „unverschlüsselt“ um

https://www.spiegel.de https://www.heise.de

lenken **automatisch** auf eine unverschlüsselte Webseite um. Der Webbrowser teilt einem nicht mit, daß man den verschlüsselten Bereich verlassen hat. Wer nicht auf das Schloß schaut wägt sich sicher, obwohl er es nicht ist!

Verwandtes..

- RootCA – Alternative Projekte
  - CaCERT (Community Projekt)
    - allerdings: RootCA ist nicht von Hause aus in den Browsern
      - daher: Warnungsmeldung
- ssh / sftp
  - Protokoll zum
    - „Einloggen“ auf einen Entfernten Computer
    - sicheren Übertragen von Dateien (Vertraulichkeit, Integrität)

..Verwandtes

- Übrigens
  - X.509-Zertifikate können
    - auch zur Verschlüsselung andere Kommunikation, z.B. E-Mail, genutzt werden (dann wird die E-Mail-Adresse im Zertifikat hinterlegt, die dann beim Prüfen beim Empfänger mit der Adresse des Absenders identisch sein muß).
    - beim E-Mail senden / abholen verwendet werden (smtps, pop3s, imaps)
    - Vergleiche auch: pgp (E-Mail, Dateiverschlüsselung; kein http)

Was bringt mir das?

- Wir haben gelernt
  - uns sicher im Netz zu bewegen
  - Vertrauenswürdigkeit einzuschätzen
  - Daten sicher zu übertragen
  - 100% sicher für alles und immer gibt es nicht
- SSL auf dem eigenen Webserver?
  - Diskussion
    - z.B. „privater Bereich“, mit .htaccess geschützt
  - Einwurf: WebDAV

Datentransfer..

Für den nächsten Vortragsblock (Webseiten) haben wir jetzt eine solide Basis erarbeitet, um zwischen den verschiedenen Protokollen zu unterscheiden, die wir verwenden können, um unsere Webseiten zu aktualisieren.

- Protokolle die einem zur Verfügung stehen sind providerabhängig
- Wir umreißen im Folgenden kurz die gängigsten Verfahren

Protokolle für den Datentransfer..

- Klassisch / obsolet
  - ftp - File-Transfer-Protocol
    - Bewertung: unsicher, weil Autorisierung mit Passworten und Datentransfer im Klartext
      - Authorisierung (Nutzername, Passwort)
      - Datentransfer (Integrität)
  - ftps
    - ftp mit ssl-Verschlüsselung
    - Bewertung: kaum verbreitet
  - telnet
    - Einloggen auf dem Server;
    - dort: Editieren der Webseiten
      - nur was für Könner
    - Bewertung: unsicher (Gründe: wie bei ftp)

..Protokolle für den Datentransfer..

- Moderner – sicherer, weil verschlüsselt
  - ssh / scp / sftp

    console
    - `ssh vortrag@kudu.in-berlin.de`
    - `scp -rpa webseiten/ vortrag@kudu.in-berlin.de:public_html/`
    - `sftp vortrag@kudu.in-berlin.de`
      - `cd public_html`
      - `put webseiten/index.html`

    GUI
    - windows: „putty“ (kostenlos)
    - MacOSX: „fugu“, „cyberduck“ (beide kostenlos)
    - Linux: in „gnome“ integriert

..Protokolle für den Datentransfer..

- WebDAV
  - Erweiterung des http-Protokolls (mehr als „PUT“, „POST“ und „GET“)
  - Autorisierung (Nutzername / Passwort)
  - Webserver sollte SSL-Zertifikat haben, damit Passworte nicht mitgelesen werden können und Daten unversehrt und verschlüsselt übertragen werden
  - Protokoll ähnlich wie „SAMBA“
    - Webserver läßt sich als „Laufwerk“ einbinden
    - Drag+Drop aus Dateimanager macht die Aktualisierung zum Kinderspiel

..Protokolle für den Datentransfer

- Unterstützte Betriebssysteme
  - Windows, MacOSX, Linux und andere

- Andere Konzepte
  - Provider hat möglicherweise ein eigenes Tool das man ansurft, und mit dem man Dateien auf den Server hochladen kann (http POST)
  - CMS statt statische Webseiten
    - **C**ontent**M**anagement**S**ysteme erlauben das Editieren und Aktualisieren von Inhalten direkt im Webbrowser
  - auch hier gilt: auf https-Verschlüsselung achten

Überleitung zum nächsten Themenblock

- Das nächste Thema ist html mit CSS
  - macht Hans
- Die Dateien liegen bereits auf dem Webserver
- Wir können sie herunterladen über
  - scp
  - webdav

..und später auf diesem Wege auch wieder hochladen zum Aktualisieren.

Zugangsdaten

- ssh / scp / sftp

  `Server: kudu.in-berlin.de`

  Nutzer: `vortrag`

  Passwort: [redacted]

- webdav

  `Server: vortrag.webdav.in-berlin.de`

  Nutzer und Passwort wie oben

EOF

Ich hoffe es war für jeden etwas interessantes dabei.

Bei Fragen: E-Mail an thomas@in-berlin.de

Die Folien finden sich auch unter

http://moabit.kiezlan.de/

und dürfen frei weitergegeben werden

Vielen Dank. für's Interesse!

privacy is not a crime ;)