CS615 - Aspects of System Administration

HTTPS, Monitoring

Department of Computer Science
Stevens Institute of Technology
Jan Schaumann
jschauma@stevens.edu
https://www.cs.stevens.edu/~jschauma/615/
HTTP

http://www.cs.stevens.edu/~jschauma/tmp/request.html
HTTP

$ sudo -v
$ sudo tcpdump -w post.pcap port 80 &
$ curl -d 'data=my-super-secret-information' \\ http://www.cs.stevens.edu/~jschauma/cgi-bin/post.cgi
$ fg
^C
$ sudo chmod a+r post.pcap

Now use `tcpdump(1)` to extract the plain text data you sent to the web server from your `pcap` file.
HTTP

IP 10.89.92.9 > 155.246.89.84: Flags [P.], seq 1:639, ack 1, length 638

HTTP/1.1 [IP: 10.89.92.9] -> [IP: 155.246.89.84]
POST /~jscha

Content-Type: application/x-www-form-urlencoded
Cookie: __cfduid=d8e04f5b4e0c570d3c43b569c061b53a1; CFPUID=666a6a613b3df49a070;

HTTP/1.1 200 OK
Content-Length: 290
Content-Type: application/x-www-form-urlencoded

POST /~jscha
HTTP

$ sudo -v
$ sudo tcpdump -w post.pcap port 443 &
$ curl -d 'data=my-super-secret-information' \  
    https://www.cs.stevens.edu/~jschauma/cgi-bin/post.cgi
$ fg
^C
$ sudo chmod a+r post.pcap

Now use tcpdump(1) to extract the plain text data you sent to the web server from your pcap file.
HTTPS

IP 155.246.89.84.443 > 10.89.92.9.50833: Flags [P.], seq 138:634, ack 1237, length 496

0x0000: 4500 0224 de34 4000 3406 0af3 9bf6 5954 E...@.4.....YT
0x0010: 0a59 5c09 01bb c691 2042 e9c5 971f 45d4 \......B...E.
0x0020: 8018 0210 0f8a 0000 0101 080a 891a 57ec ..............W.
0x0030: 3d7d 29d4 1703 0301 0515 a4d7 9c25 9a45 =v)......%.E
0x0040: 653d ee2c d8d7 d53e 045f a778 5cac e270 e=,......x...
0x0050: 7d78 0e00 e240 e240 3406 0af3 9bf6 5954 }...E.4...YT
0x0060: f40b c972 e95f 6df7 710 230f 4b54 (.x...g...#KT
0x0070: e675 5dbd 7cc7 b00a 49cd 645a 0e7c 4cf8 u[...I.dZ.L
0x0080: 7120 dc31 d1e5 b3f4 5b5c 6e57 e43c f6aa q...1...[w.<.
0x0090: 7499 604d c6e6 0152 098e 3fca 66ac 5929 t.'F...?...f.Y
0x00a0: 5777 6c2f 2658 eca1 5fa6 3ef6 476f 42fe Wwl/&X.._.>GoB.
0x00b0: cb26 4948 4194 f23a ced9 2a67 cf7d bcc3 "IHA...*g."
0x00c0: 2046 ad15 233c ff22 3303 0095 15a4 d79c .F..<3!...B3
0x00d0: 515e be8f 03c0 786b f066 bece f961 7996 Q"..xk....ay.
0x00e0: f352 6a1c 0968 726e 819a c927 2669 358c .Rj...hnr...'15.
0x00f0: 9c30 e8a8 6ccf d30c 4bfc e689 7a8f 6ec4 0.1...K...zn.
0x0100: f232 9c14 6394 39f1 56e6 3e8a c910 e8b4 .2.c.9.V...>
0x0110: 79c8 44ca dde0 86cc 3a4a e4c4 ec15 1703 y.D......J.
0x0120: 0300 2215 ad47 9c25 9a45 66b1 c56f b2c4 ."...%.Ek...o.
0x0130: de96 6808 09b6 b553 9de1 c6d6 9adc cb99 ..h...S...n.
0x0140: 9099 642e 1817 0303 0095 15a4 d79c 259a ..d........%.
0x0150: 4567 617a 87fa e66d cef1 c2f0 6101 a7dd Egaz...m...a.
0x0160: bfbe 756b cc50 26fb af35 1ffe e842 c1cc ...uk.P&.5...B.
0x0170: 5bae cc33 3110 ac66 bf43 7897 fad8 5e80 [.31.f.Cx...`
0x0180: 509e 7305 e58b 1aaf 0e96 76e0 aa24 f900 P.s......v.$
0x0190: 290a 9260 6052 6ac0 6b3c f8c6 f873 8fbb )"(Rjk....s.
0x01a0: af6f ee9c 0a35 7e9c ca18 7ad7 9cd9 e2cc .o...5"...z....
0x01b0: 8cec 4034 4970 bf94 44ce 0adb 3778 7648 ..@4Ip.L...7xvH
0x01c0: 10c7 3505 09fd ff80 fe27 7b1d 34ac c066 ...5......'{.4..f

HTTPS, Monitoring March 20, 2017
HTTPS

HTTPS stands for...

HTTP over SSL.
HTTPS

HTTPS stands for...

HTTP over SSL.

HTTP over TLS.
HTTPS

HTTPS stands for...

HTTP over SSL.

HTTP over TLS.

Secure HTTP.
HTTPS

HTTPS stands for...

HTTP over SSL.

HTTP over TLS.

Secure HTTP.

HTTP Secure.
HTTPS

HTTPS stands for...

HTTP over SSL.

HTTP over TLS.

Secure HTTP.

HTTP Secure.

But it uses TLS. And used to use SSL. Although hopefully not any more. Although probably still.

SSL is dead. Don’t use it. Seriously, don’t.

We should really only call it TLS. HTTPT.
TLS

CS615 - Aspects of System Administration

March 20, 2017
TLS

Transport Layer Security
- set of cryptographic protocols
- operates on layer 6 of OSI stack (Presentation Layer) (or 5? 4? 7? none? all?)
- independent of HTTP
- RFC5246 (TLS 1.2)

Two distinct security mechanisms:
1. encryption of data in transit
2. authentication of parties
TLS

Protocol:
- Client Hello, present list of supported cipher suites
- Server Hello, chosen cipher suite
- Server Certificate
- (Server Key Exchange Message), (Client Certificate Request), (Client Certificate)
- Client Key Exchange Message
- (Certificate Verify)
- (Client Change Cipher Spec), (Server Change Cipher Spec)
### TLS

#### Handshake Protocol: Client Hello
- **Handshake Type:** Client Hello (1)
- **Length:** 512

#### Cipher Suites
- **Cipher Suite:** TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0x0303)
- **Cipher Suite:** TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc010)
- **Cipher Suite:** TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc002)

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.000000</td>
<td>166.84.7.99</td>
<td>155.246.89.84</td>
<td>TCP</td>
<td>78</td>
<td>57691-443 [SYN] Seq=3580837243 Win=32768 Len=0 MSS=1460 WS=0 _</td>
</tr>
<tr>
<td>2</td>
<td>0.004559</td>
<td>155.246.89.84</td>
<td>166.84.7.99</td>
<td>TCP</td>
<td>74</td>
<td>74 443-57691 [SYN, ACK] Seq=2770027821 Ack=3580837244 Win=14498 _</td>
</tr>
<tr>
<td>3</td>
<td>0.004601</td>
<td>166.84.7.99</td>
<td>155.246.89.84</td>
<td>TCP</td>
<td>66</td>
<td>57691-443 [ACK] Seq=3580837244 Ack=2770027822 Win=33576 Len=0 _</td>
</tr>
<tr>
<td>4</td>
<td>0.008020</td>
<td>166.84.7.99</td>
<td>155.246.89.84</td>
<td>TLSv1.2</td>
<td>583</td>
<td>Client Hello</td>
</tr>
<tr>
<td>5</td>
<td>0.013809</td>
<td>155.246.89.84</td>
<td>166.84.7.99</td>
<td>TCP</td>
<td>66</td>
<td>443-57691 [ACK] Seq=2770027821 Ack=3580837761 Win=15552 Len=0 _</td>
</tr>
<tr>
<td>6</td>
<td>0.021535</td>
<td>155.246.89.84</td>
<td>166.84.7.99</td>
<td>TLSv1.2</td>
<td>15_ Server Hello</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0.022047</td>
<td>155.246.89.84</td>
<td>166.84.7.99</td>
<td>TCP</td>
<td>15_ [TCP segment of a reassembled PDU]</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.022667</td>
<td>155.246.89.84</td>
<td>155.246.89.84</td>
<td>TCP</td>
<td>66</td>
<td>57691-443 [ACK] Seq=3580837761 Ack=2770030718 Win=32128 Len=0 _</td>
</tr>
<tr>
<td>9</td>
<td>0.022615</td>
<td>155.246.89.84</td>
<td>155.246.89.84</td>
<td>TLSv1.2</td>
<td>13_ Certificate</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>0.028234</td>
<td>155.246.89.84</td>
<td>155.246.89.84</td>
<td>TLSv1.2</td>
<td>192 _ Client Key Exchange, Change Cipher Spec, Encrypted Handshake _</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>0.033376</td>
<td>155.246.89.84</td>
<td>166.84.7.99</td>
<td>TLSv1.2</td>
<td>117 Change Cipher Spec, Encrypted Handshake Message</td>
<td></td>
</tr>
</tbody>
</table>

---

**HTTPS, Monitoring**

*March 20, 2017*
$ openssl s_client -connect www.cs.stevens.edu:443

[...]
New, TLSv1/SSLv3, Cipher is DHE-RSA-AES256-SHA
Server public key is 2048 bit
Secure Renegotiation IS supported
Compression: NONE
Expansion: NONE
SSL-Session:
  Protocol : TLSv1
  Cipher : DHE-RSA-AES256-SHA
  Session-ID: 5F8A9B7A93EF87009EFCC17BBDD8938C56EAACD9DF4C3643EF034D0479F44C9
  Session-ID-ctx:
  Master-Key: 20CBA1E477A8B573F29759045329EF7AA38C763C4C41606A46FBCC824C3F32F708789
  Key-Arg : None
  Start Time: 1460395966
  Timeout : 300 (sec)
  Verify return code: 0 (ok)
$ openssl s_client -connect www.cs.stevens.edu:443 | \
  openssl x509 -text -noout

[...]
Signature Algorithm: sha256WithRSAEncryption
Issuer: C=US, ST=MI, L=Ann Arbor, O=Internet2, OU=InCommon, CN=InCommon RSA Server
Validity
  Not Before: Mar 3 00:00:00 2017 GMT
  Not After : Mar 2 23:59:59 2020 GMT
Subject: C=US/postalCode=07030, ST=NJ, L=Hoboken/street=1 Castle Point on Hudson
Subject Public Key Info:
  Public Key Algorithm: rsaEncryption
  RSA Public Key: (2048 bit)

[...]
X509v3 Subject Alternative Name:
  DNS:www.cs.stevens.edu, DNS:rcs.srcit.stevens.edu, DNS:svn.srcit.stevens.edu
  DNS:www.srcit.stevens.edu

Note the absence of 'stevens-tech.edu' names...
### TLS

Setting up a Man in the Middle attack site:

1. start instance

2. openssl req -x509 -nodes -days 365 -sha256 -newkey rsa:2048 -keyout mycert.pem -out mycert.pem

3. sudo openssl s_server -WWW -accept 443 -cert mycert.pem

4. curl https://www.stevens.edu/sit/ > index.html

4. go to https://<instance>/
TLS Authentication

Use of X.509:

- public key certificates
- certificate revocation lists (CRLs) / Online Certificate Status Protocol (OCSP)
- certificate path validation under a Public Key Infrastructure (PKI)
- certificate chains depend on trust anchors
TLS

1. User / Company generates a Certificate Signing Request (CSR), containing:
   - identifying information (distinguished name etc.)
   - signature of data by private key
   - chosen public key
TLS

1. User / Company generates a *Certificate Signing Request* (CSR)

2. CSR submitted to Certificate Authority (CA)
TLS

1. User / Company generates a *Certificate Signing Request* (CSR)

2. CSR submitted to Certificate Authority (CA)

3. CA verifies information
TLS

1. User / Company generates a *Certificate Signing Request (CSR)*

2. CSR submitted to Certificate Authority (CA)

3. CA verifies information

4. CA returns certificate signed with its private key
TLS

1. User / Company generates a Certificate Signing Request (CSR)

2. CSR submitted to Certificate Authority (CA)

3. CA verifies information

4. CA returns certificate signed with its private key

5. clients can verify signatures against trusted root CAs
CS615 - Aspects of System Administration

TLS

User generates key pair, KEpub, KEpri.
User generates CSR, containing CN=www.example.com, KEpub, signed with KEpri.
User submits CSR to CA.
CA verifies User / ownership of www.example.com.
CA issues certificate, containing CN=www.example.com, KEpub, signed with KEpri.
User installs certificate, KEpri, intermediate on server(s).
Client requires CA's root cert to be installed and trusted.
Client connects to www.example.com.
Server presents certificate(s).
Client verifies signature on certificate was made with KEpri, signature on intermediate was made with KERootpri.

This diagram is in the public domain. Originally made by @jschauma.
TLS Pitfalls

195 root CAs on this laptop...
TLS Pitfalls

Lack of universal HTTPS exposes users to significant risks; many sites don’t understand the importance of authentication for non-sensitive content.

In order to serve content, you need to have the private key => privkey available at perimeter and exposed, high-risk systems.

Rotation/renewal of keys requires routine processes, which may further expose the private key.

Control of a CA or a CA’s key grants you near universal powers.
TLS Pitfalls

Complex protocols, buggy implementations, intentional weaknesses and backwards compatibility are just the high level points.

- SSLv2 obsoleted in 1996; 2016: DROWN attack
- SSLv3 obsoleted in 1999; 2014: POODLE attack
- BEAST, CRIME, BREACH, HEARTBLEED, GotoFail...
- Obsolete and broken algorithms widely used (RC4, MD5, SHA1, ...)

HTTPS, Monitoring
TLS

Additional related topics:

- HSTS and TLS stripping attacks
- HPKP and Trust On First Use (TOFU)
- Content Security Policy (CSP)
- “Secure” cookies vs. HttpOnly cookies
- attacks on domain name registrars

Security is difficult. More on that in a future lecture.
Hooray!

5 minute break
Problem Report

“Something’s wrong.”
Now what?
Problem Report

“The system feels slow.”

“I can’t log in.”

“My mail was not delivered.”

“The site is down.”
Now what?
To the logs!
Answers

“The system feels slow.”
up 1318 days, 13:46, 1 user, load averages: 993.81, 272.91, 1012.18

“I can’t log in.”
Apr 6 09:25:56 <auth.info>hostname sshd[1624]: Failed password for jdoe from 115.239.231.100 port 1047 ssh2

“My mail was not delivered.”
Apr 11 16:15:40 panix postfix/smtpd[7566]: connect from unknown[122.3.68.122]
Apr 11 16:15:41 panix postfix/smtpd[7566]: NOQUEUE: reject_warning: RCPT from unknown[122.3.68.122]: 450 4.7.1 Client host rejected: cannot find your hostname, [122.3.68.122]; from=<McneilRomany28@pldt.net> to=<jschauma@stevens.edu> proto=ESMTP helo=<122.3.68.122.pldt.net>
Answers

“The site is down.”

403 524 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:28.0)
Gecko/20100101 Firefox/28.0"
Answers

"The site is down."

94.242.252.41 - "" [11/Apr/2016:19:18:47 -0400] "GET /secret/ HTTP/1.1" 403 524 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:28.0) Gecko/20100101 Firefox/28.0"
“Something’s wrong.” is just an *unexpected* or *undesirable* event.
“Something’s wrong.” is just an *unexpected* or *undesirable* event.

*Events* happen all the time.
“Something’s wrong.” is just an *unexpected* or *undesirable* event.

*Events* happen all the time.

Being able to identify *relevant* events allows you to diagnose, predict and even prevent *undesirable* events.
Events

In order to be able to identify an event as *unexpected*, you have to have *expected* events.
Expected Events

Know your applications.
Expected Events

Know your applications.

Know your users.
Expected Events

Know your applications.

Know your users.

Know your traffic patterns.
Expected Events

- Know your applications.
- Know your users.
- Know your traffic patterns.

*Know your systems.*
Events and Metrics

$ dict event
  event
    n 1: something that happens at a given place and time
    2: a special set of circumstances; "in that event, the first possibility is excluded"; "it may rain in which case the picnic will be canceled" [syn: {event}, {case}]

$ dict metric
  metric
    3: a system of related measures that facilitates the quantification of some particular characteristic [syn: {system of measurement}, {metric}]
Events and Metrics

Event  Metric  You
Events and Metrics

Events
- may occur rarely / frequently / constantly
- can be collected in logs
- may be comprised of other events
- may be: something happened
- may be: nothing (new) happened

Metrics:
- correlation of related events
- may help identify outliers
- may trigger events
- may help make (automated or interactive) decisions
Collecting Data

*Counters*: easy, numeric data tracking individual events. Example: HTTP status codes

*Timers*: easy, numeric data tracking event duration. Example: Time to send all data for a successful HTTP request.

*Thresholds*: easy, numeric trigger for events; may itself trigger events or metrics. Example: more than N HTTP hits in X seconds yield 404.
Know Your Systems

Profile your application:
- execution time (for example: `time(1)`) 
- data sources and destination affect execution
- `strace(1)` and friends for more detailed analysis

Understand your system performance:
- CPU load, memory (for example: `top(1), vmstat(1)`) 
- disk I/O (for example: `iostat(1)`) 
- user activity (for example: `ac(1), lsof(8), sa(8)`)
Know Your Systems

Network statistics:

- ports and applications (for example: `lsof(8), netstat(8)`)
- packets in and out
- connection origin
- *NetFlow* etc.
Context lets you find relevant events in your haystack of metrics.
CPU load - 12 hours

- **idle**: min=51.14 max=72.14
- **user**: min=15.66 max=16.87
- **system**: min=11.36 max=13.05
- **interrupt**: min=0.01 max=0.26
- **nice**: min=0.00 max=19.43

HTTPS, Monitoring

March 20, 2017
Disk I/O - 12 hours

- Reads: min=0, max=336, average=27 transactions/sec
- Writes: min=28, max=55, average=33 transactions/sec
- Busy count: min=0, max=0, average=0 transactions/sec
Load Average - 12 hours

![Graph showing load average over 12 hours]

- 1 min: min=1.018, max=2.085
- 5 min: min=1.048, max=1.956
- 15 min: min=1.062, max=1.865

HTTPS, Monitoring

March 20, 2017
No context.

Memory - 12 hours

- total memory: min=8115504 max=8115504
- free memory: min=77002 max=6661555
- active memory: min=46317 max=52116
- inactive memory: min=1008720 max=7278141
- wired memory: min=374892 max=561227
- cached memory: min=9033 max=332417

HTTPS, Monitoring
March 20, 2017
Some context.

12 hours

[Graphs showing system metrics over 12 hours]
With context.

7 days
Know your systems.

CPU load - 30 days
Know your systems.

30 days

HTTPS, Monitoring March 20, 2017
Turn *events* into *metrics*.

- Log it!
- Export counters/timers from within your application.
- Process logs and produce counters/timers:
  ```
  awk {print $9} /var/log/httpd/access.log | sort | uniq -c
  ```
- Graph it.
  ```
  http://shouldigraphit.com/
  ```
Monitoring/graphing

SNMP based:
- Cacti: http://www.cacti.net/
- MRTG: http://oss.oetiker.ch/mrtg/
- Observium: http://demo.observium.org/
- ...

Other / complementary:
- Ganglia: http://monitor.millennium.berkeley.edu/
- Munin: http://munin.ping.uio.no/
- Nagios: http://nagioscore.demos.nagios.com/
- Graphite: http://graphite.wikidot.com/
To the cloud!

There's a service for that. In the cloud.

Consider:
- support / convenience vs. do-it-yourself
- integration with your other services
- data confidentiality
- data lock-in (esp. when trending data over years)
Increasing the size of your haystack does not always help in finding the needle.
Monitoring Pitfalls

Increasing the size of your haystack does not always help in finding the needle.

Email is not a scalable network monitoring solution.
Increasing the size of your haystack does not always help in finding the needle.

Email is not a scalable network monitoring solution.

Absence of a signal can itself be a signal.
Monitoring Pitfalls

Increasing the size of your haystack does not always help in finding the needle.

Email is not a scalable network monitoring solution.

Absence of a signal can itself be a signal.

This list is incomplete.
Reading

HTTPS / TLS:

- RFC5246 (TLS 1.2) and RFC6176 (prohibiting SSL)
- https://bugzilla.mozilla.org/show_bug.cgi?id=647959
- https://cabforum.org
Reading

Monitoring:

- http://www.datadoghq.com/
- https://www.newrelic.com/
- http://logstash.net/
- http://www.splunk.com/