tcpcrypt

Andrea Bittau, Michael Hamburg, Mark Handley, David Mazières, Dan Boneh

Stanford University, University College London
Goal: encrypt most TCP traffic

- Zero configuration, works with NATs.
- Integrate with app-level authentication
- High performance, especially on servers.
- Avoid double encryption.
## Maximize security for each scenario

<table>
<thead>
<tr>
<th>Use case</th>
<th>Preconfiguration</th>
<th>Today’s security</th>
<th>Possible security</th>
</tr>
</thead>
<tbody>
<tr>
<td>News site</td>
<td>None</td>
<td>None</td>
<td>No passive eavesdropping</td>
</tr>
<tr>
<td>Online shop</td>
<td>Server certificate</td>
<td>Server auth</td>
<td>Server auth</td>
</tr>
<tr>
<td>Forum</td>
<td>Shared secret (cookie) no server certificate</td>
<td>None</td>
<td>Mutual auth</td>
</tr>
<tr>
<td>Banking</td>
<td>Shared secret and server certificate</td>
<td>Mutual auth if cert and pass OK</td>
<td>Mutual auth if pass OK</td>
</tr>
</tbody>
</table>
tcpcrypt handshake

SYN
tcpcrypt HELLO option

SYN ACK
PKCONF: public key ciphers, key sizes

ACK
INIT1: symmetric ciphers, MACs, nonce, public key

ACK
INIT2: encrypted client and server nonce (master key)

INIT1/2 don’t fit in SYN / ACK: sent as data invisible to apps.
Session cached handshake

Master key

Encryption and MAC keys

Session ID

Next connection

Next key

SYN

tcpcrypt NEXTK1

SYN ACK

NEXTK2

ACK

Low latency!
### MAC and Encryption

<table>
<thead>
<tr>
<th>src port</th>
<th>dst port</th>
</tr>
</thead>
<tbody>
<tr>
<td>seq no</td>
<td>(64-bit seq)</td>
</tr>
<tr>
<td>ack no</td>
<td>(64-bit ack)</td>
</tr>
<tr>
<td>d. off.</td>
<td>window</td>
</tr>
<tr>
<td>flags</td>
<td>checksum</td>
</tr>
<tr>
<td>window</td>
<td>urg ptr</td>
</tr>
<tr>
<td>options (e.g., SACK)</td>
<td>MAC option</td>
</tr>
<tr>
<td>data</td>
<td>TCP length</td>
</tr>
</tbody>
</table>

- **d. off.** (data offset)
- **flags**
- **window**
- **checksum**
- **urg ptr** (urgent pointer)
- **options (e.g., SACK)**
- **MAC option**
- **data** (64-bit seq and 64-bit ack)
- **TCP length**

**MACed**

**MACed & Encrypted**
tcpcrypt semantics

getsockopt(s, SOL_TCP, TCP_SESSIONID, &sid, sizeof(sid));

- If session ID is equal on both endpoints, no man in the middle.
- Authenticating session ID authenticate connection:
  - E.g., sign SID with cert, HMAC with cookie, password (PAKE), ...
  - Can also log and check after the fact.
High performance

• Up to 25x higher connection accept rate than SSL on servers.

• Near TCP connection latency for session cached tcpcrypt connections.

• 9Gbit/s using AES+UMAC with AES-NI
Conclusion

• Encryption is general-purpose and practical to enable by default.

• Benefits of encrypting at transport layer:
  • Backwards compatible (e.g., NATs).
  • Benefits legacy apps.
  • Natural granularity for authentication.
  • Leverage existing handshake for negotiation.

http://tcpcrypt.org