SOCKS Protocol Version 6 (Update)
draft-olteanu-intarea-socks-6-02

Vladimir Olteanu, Dragoș Niculescu
University Politehnica of Bucharest
Overview

• 0-RTT overhead and TFO support
  – Clients optimistically send as much information upfront
  – 0-RTT authentication

• Run over TLS (protect against malicious 3rd parties)
  – Mitigate early data replay attacks
  – Plaintext password authentication now viable

• setsockopt()-like mechanism (new in -02)
  – MPTCP scheduler
  – Discovery of servers supporting MPTCP (for proxy bypass)
SOCKSv5 vs. SOCKSv6
SOCKSv5 vs. SOCKSv6
Plain text password authentication

• Viable if done over TLS
  - Expected de facto standard
• Initial message from RFC1929 placed in SOCKS Request as an option
  - 0 RTT
  - Only if it fits: ULEN + PLEN <= 249

+---------------+--------------+----+------+----------+----------------+----------+
| Kind | Length | Method = 0x2 | VER | ULEN | UNAME   | PLEN | PASSWD |
+------+--------+--------------+----+------+----------+------+----------+
| 1    | 1      | 1            | 1  | 1    | 1 to 255 | 1    | 1 to 255 |
Socket Options

- Part of Requests and Operation Replies
- Inspired by setsockopt()/getsockopt() (from *nix)
  - Not an RPC
  - Individual options must be standardized separately
- Will be renamed in -03

<table>
<thead>
<tr>
<th>Kind</th>
<th>Length</th>
<th>Leg</th>
<th>Level</th>
<th>Code</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2 bits</td>
<td>6 bits</td>
<td>1</td>
<td>Variable</td>
</tr>
</tbody>
</table>

- Leg: Client-Proxy (0x1), Proxy-Server (0x2) or Both(0x3)
- Level: Socket, IPv4, IPv6, TCP, UDP
- Code
TFO Option

• Replaces field in Request
• As part of a CONNECT Request: TFO SHOULD be attempted
  – Absence means TFO MUST NOT be attempted
• As part of an Operation Reply: TFO succeeded

+---------------+--------+--------+------+
| Kind | Length | Leg   | Level | Code |
+------+--------+--------+--------+------+
|  1   |   1    | 2 bits | 6 bits |  1   |
+------+--------+--------+--------+------+

• Leg: Proxy-Server (0x2)
• Level: TCP
• Code: 0x17
Proxy Bypass

- Let multihomed clients know when a server supports MPTCP
  - Can contact server directly
- Place MPTCP option in Operation Reply

<table>
<thead>
<tr>
<th>Kind</th>
<th>Length</th>
<th>Leg</th>
<th>Level</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2 bits</td>
<td>6 bits</td>
<td>1</td>
</tr>
</tbody>
</table>

- Leg: Proxy-Server (0x2)
- Level: TCP
- Code: 0x17
Proxy Bypass

• Let multihomed clients know when a server supports MPTCP
  – Can contact server directly
• Place MPTCP option in Operation Reply

<table>
<thead>
<tr>
<th>Kind</th>
<th>Length</th>
<th>Leg</th>
<th>Level</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2 bits</td>
<td>6 bits</td>
<td>1</td>
</tr>
</tbody>
</table>

• Leg: Proxy-Server (0x2)
• Level: TCP
• Code: 0x17
Choosing the MPTCP Scheduler

- As part of a Request: indicates the scheduler to be used
- As part of an Operation Reply: indicates what scheduler is used
- Supports schedulers available in the Linux MPTCP implementation
- Use case: low latency services
  - The REDUNDANT scheduler duplicates data across paths

<table>
<thead>
<tr>
<th>Kind</th>
<th>Length</th>
<th>Leg</th>
<th>Level</th>
<th>Code</th>
<th>Scheduler</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>6</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

- Level: TCP
- Code: 0x2b
- Scheduler: Default/Round-Robin/Redundant
Backup Slides
Salt Options

- Clients may make multiple duplicate requests
  - May be encrypted using the same PSK
- Intended to protect against profiling attacks by adding a random value
  - TLS 1.3 forces everyone to use AEAD
  - Salt option is redundant; will remove in -03