A System for Authenticated Policy-Compliant Routing

Barath Raghavan and Alex C. Snoeren
UC San Diego
Routing Today

• ISPs perform wide-area routing through BGP
  – Used to express local policy and traffic eng.
  – Problem: users can’t express routing preferences
• Overlay routing / IP source routing
  – Enables edge routing control
  – Allows pooling of resources
  – Problem: may interfere with ISP policy and traffic engineering
Our system: Platypus

• Loose source routing in which…
  – Users can pick their routes
  – ISPs control placement of indirection points
• … and authentication which enables…
  – ISPs to verify the policy-compliance of traffic
  – Is easily accountable
  – Delegation of source routing rights by users
An example

Local policy avoids customer route through C

Default route

Optimal route

C could forward traffic
The challenge

How can C provide forwarding service?
Our system: Platypus

Packet explicitly sent through \( C \)

1. Negotiate contract
2. Receive source routing info
3. Stamp + send packets

Indirection point to route through
Key building blocks

- Routing system providing basic connectivity
- Path discovery mechanisms / services
- Negotiation of business relationships
- Mechanism for authenticated loose source routing
Network Capabilities

• Specify a hop of the source route, including:
  – Point of indirection, called a *waypoint*
  – The responsible party, called the *resource principal*

• Waypoints are:
  – Chosen by ISPs
  – Specified by a routable IP address
Authentication

Requires asymmetry of information: H1 must know more than H3

Goal: Distinguish between valid and invalid packets
Authentication keys

• Each waypoint has one waypoint key $k$
• Each resource principal has a secret key $s$
  – Derived from waypoint key using a keyed MAC
  – Unique given a waypoint and a capability
Packet Stamping

- IP Header
- Platypus Header
- Payload

Waypoint ID
Resource Principal ID
Auth Info (Binding)
Capabilities

Invariant headers + payload
Secret $s$

MAC
Packet Verification

- IP Header
- Platypus Header
- Payload
- Waypoint key $k$
- Capability $c$
- Header+payload
- MAC
- Temporal secret $s$
- Binding $b$
- Packet binding $b'$
- Forward
Temporal secrets

• Temporal secret keys expire periodically
  – Expiration allows for changing policies

• No time sync required
  – Secret computation includes Key ID/time
  – Enables expiration on order of clock drift

• Requires lookup of temporal secrets
Key lookup

- DNS-based key lookup
  - DNS reply contains encrypted secret
  - No key distribution infrastructure required
  - Key lookup as fast as DNS lookup
Delegation

• Users may pass out their capabilities
  – How might they restrict others’ use?

• Capability delegation:
  – Principals can restrict capabilities
  – Limits holder to destinations within an IP prefix
  – Useful to ensure similar reverse paths

Undesirable asymmetry
Implementation

- End-host based stamping/forwarding
- User-level and kernel module versions
Per-packet latency

- Total per-packet time = I/O time + header processing
- I/O time ~ 2 \( \mu \)s
- Worst-case header processing time < 2 \( \mu \)s

<table>
<thead>
<tr>
<th>Header processing overhead</th>
<th>68 byte</th>
<th>348 byte</th>
<th>1500 byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>Null</td>
<td>172 ns</td>
<td>173 ns</td>
<td>181 ns</td>
</tr>
<tr>
<td>UMAC</td>
<td>695 ns</td>
<td>998 ns</td>
<td>1908 ns</td>
</tr>
</tbody>
</table>
Deployment

• Incrementally deployable
  – Does not require inter-ISP cooperation
  – *Loose* source-routing based

• How might ISPs deploy Platypus?
  – Where should they be placed?
  – How many Platypus waypoints are needed?
Measurement study
Waypoint effectiveness (MCI)

![Waypoint effectiveness graph](graph.png)
Summary and future work

• Platypus provides:
  – Source routing with ISP control of waypoints
  – Means for authenticating source routed packets

• Incremental deployment
  – Flow-based Platypus with existing hardware

• New forwarding business model
  – Anyone can sell/resell forwarding service
  – Real-time pricing of capabilities
Scalability

• Forwarding state
  – Waypoints only need $O(1)$ state

• Key lookup
  – Lookup overhead is small (3 crypto operations)
  – One key server $\sim 500,000$ lookups / sec

• Per-principal accounting
  – High speed approx. per-flow counters [Kumar ’04]
## Platypus header format

<table>
<thead>
<tr>
<th></th>
<th>Flags</th>
<th>Capability List Length</th>
<th>Capability List Pointer</th>
<th>Encapsulated Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Original Source Address</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final Destination Address</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Waypoint Address</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Resource Principal</td>
<td>Key ID</td>
<td>Flags</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4 bytes
Temporal secret computation

- For a capability $c$ and waypoint key $k$:
  $$s = \text{MAC}_k(c\_\text{way} || c\_\text{rp} || ((t\gg n) \& 0xFFFFFFFF0) | c\_\text{id})$$
- The exception to this is at key ID wraparound
  - $(t\gg n)$ is either incremented or decremented by 1
Measurement results (QWEST)
Measurement results (GBLX)
Measurement results (SPRINT)
Example: Virtual multihoming

Using Platypus, C can *virtually multihome* with ISPs 1 and 3.
Using Platypus, $C$ can distribute delegated capabilities that are restricted to send to prefixes within $C$. 

Example: Affecting Inbound Traffic