Securing BGP
Large scale trust to build an Internet again

Lutz Donnerhacke

db089309: 1c1c 6311 ef09 d819 e029 65be bfb6 c9cb
A protocol from better times

• A protocol from the early Internet
  • People were friendly and trustworthy
  • Internet was a warm and fuzzy place

• BGP: protocol from admins for managers
  • Main assumption: Routers do not lie
  • Idea 1: Announce what you have
  • Idea 2: Redistribute politically

• Inject locally, route globally
An example
Policy documentation

- Whois database
  - Distributed store of resource allocation
  - Database ensures correctness

- RPSL database
  - Centralized store of peering information
  - Both views of a peering: Sender / Receiver
  - Detailed peering policy incl. filter, precedence

- Software available
  - Generates router configuration
Threats to BGP

• Fat fingers
  • Announcing wrong network
  • Prepending foreign ASN
• Broken devices
  • Bitflip in memory or transit
• Commercial/criminal attacks
  • Redirect traffic (claim prefix, claim peering)
  • Inject unallocated networks (sending Spam)
• Governmental/Lawful attacks
  • Filtering traffic to protect the innocent
soBGP

- Trustworthy ISP approach
  - Transport authorisations as BGP attribute
  - Certifying assignment of a prefix by parent
- Each AS is a X.509-CA
  - Certifying injection policy per prefix (which ASNs are/is/isn’t the first peerings)
  - Certifying it own peering policy with peers
- Web of trust
  - Resilience against erroneous behaviour
  - Permitting multiple hierarchies
S(ecure)-BGP

• RPKI approach
  • Transport authorisations as BGP attribute
  • Certifying allocation of prefix/ASN top-down
• Each ISP is a X.509-CA
  • Certifying injection policy: Prefixes per ASN
  • Certifying it own routers to sign redistribution
• Trust anchor management
  • Accessing various CA repositories
S-BGP operation

• Routers
  • Access external caches for object verification
  • Sign each update announcement
  • New hardware for storage and crypto operation

• Resource deallocation
  • Prefix updates time out => ~15 updates/s
  • Certificate and CRL times out => rsync

• Only one structure
  • Errors are disastrous
  • Ideal for LE
An other approach

- RPSL / Whois
  - Use it for non-local checks (was it allowed?)
  - No modification to BGP protocol
  - Skips gaps in deployment
  - Fails to deal with non-public policies
- Use DNSSEC?
  - DNS as a trustworthy, distributed database
  - Routers: Offload crypto to AD-bit, caching implicit
  - Drastic RPSL simplification necessary
## Comparison

<table>
<thead>
<tr>
<th>Criteria</th>
<th>soBGP</th>
<th>Secure-BGP</th>
<th>RPSL</th>
<th>DNSSEC</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASN Alloc</td>
<td>Web of trust</td>
<td>RPKI</td>
<td>Whois</td>
<td>DNS</td>
</tr>
<tr>
<td>Prefix Alloc</td>
<td>Web of trust</td>
<td>RPKI</td>
<td>Whois</td>
<td>DNS</td>
</tr>
<tr>
<td>Private IP/AS</td>
<td>Other TA</td>
<td>Other TA</td>
<td>No</td>
<td>Stub zone</td>
</tr>
<tr>
<td>Router in AS</td>
<td>Validated</td>
<td>Validated</td>
<td>Unchecked</td>
<td>Unchecked</td>
</tr>
<tr>
<td>Outgoing Peer</td>
<td>Validated</td>
<td>Traced</td>
<td>Validated</td>
<td>Existence</td>
</tr>
<tr>
<td>Incoming Peer</td>
<td>Validated</td>
<td>Unchecked</td>
<td>Validated</td>
<td>Existence</td>
</tr>
<tr>
<td>Withdraw</td>
<td>Unchecked</td>
<td>Unchecked</td>
<td>Validated</td>
<td>Validated</td>
</tr>
<tr>
<td>Early scope</td>
<td>Many islands</td>
<td>Few islands</td>
<td>Full network</td>
<td>Full network</td>
</tr>
<tr>
<td>BGP protocol</td>
<td>Change</td>
<td>Change</td>
<td>Keep</td>
<td>Keep</td>
</tr>
<tr>
<td>Router HW</td>
<td>Change</td>
<td>Change</td>
<td>Keep</td>
<td>Keep</td>
</tr>
<tr>
<td>Helper Device</td>
<td>No</td>
<td>Simple Cache</td>
<td>Complex API</td>
<td>Resolver</td>
</tr>
</tbody>
</table>
Questions?
Why the approach is wrong
Why the approach is still wrong