BGP Route Security
Cycling to the Future!

Alexander Azimov
Yandex mitradir@yandex-team.ru
Malicious Hijacks/Leaks

- FISHING SITES
- HIJACK OF HTTPS CERTIFICATES
- SPAM/BOTNET ACTIVITY
- DOS ATTACKS
Hijack Factory Shutdown (2018)

• 25 June – first report on NANOG mailing list;
• 30 June – disconnect from HE;
• 07 July – disconnect from IXes;
• 15 July – disconnect from Cogent;
• 23 July – disconnect from GTT;

Win!!!
BGP Hijack Factory Shutdown

• 25 June – first report on NANOG mailing list;
• 30 June – disconnect from HE;
• 07 July – disconnect from IXes;
• 15 July – disconnect from Cogent;
• 23 July – disconnect from GTT;

Win!!! But does it scale?
We are always inventing new bicycles!
IRR Filters

bgpq3 -S ripe as-yandex | head

no ip prefix-list NN
ip prefix-list NN permit 2.60.0.0/14
ip prefix-list NN permit 2.60.0.0/16
ip prefix-list NN permit 2.61.0.0/16
ip prefix-list NN permit 2.62.0.0/16
ip prefix-list NN permit 2.62.0.0/17
ip prefix-list NN permit 2.63.0.0/17
ip prefix-list NN permit 2.63.0.0/18
ip prefix-list NN permit 2.63.64.0/18
ip prefix-list NN permit 2.72.0.0/13
IRR Filters

User Wins!

Route: x.x.x.x
AS_PATH: AS4

Route object (AS1, x.x.x.x)
IRR Filters: Bypassed

Attacker Wins!

Route Object (AS1, x.x.x.x)
Route object (AS4, x.x.x.x)

Route: x.x.x.x
AS_PATH: AS4
Key Findings: IRR Filters

IRR Filters Can be Used to:
• Filter **some mistake** hijacks;
• Filter **some mistake** route leaks.

IRR Filters Can’t be Used to:
• Filter malicious activity

In reality:
• Many AS-SETs are **poorly maintained**;
• **No filters** at some huge Tier-2 networks;
• Even some **Tier1 networks fail** to configure filters;

Source: [https://ripe76.ripe.net/presentations/37-ripe76.azimov.pdf](https://ripe76.ripe.net/presentations/37-ripe76.azimov.pdf)
IRRrrrrr!
ROA Validation
ROA Validation (prefix, ASN)

1. Retrieve all cryptographically valid ROAs in a for selected *prefix*. This selection forms the set of candidate ROAs.

2. If the set of candidate ROAs is empty, then the procedure exits with an outcome of unknown.

3. If there is at least one candidate ROA where the AS number is ASN and prefix length less or equal to max_length option then the procedure exits with an outcome of valid.

4. Otherwise, the procedure exits with an outcome of invalid.
ROA Validation

User Wins!

Route: x.x.x.x
AS_PATH: AS4

ROA {x.x.x.x, AS1}
ROA Validation: Bypassed

Attacker Wins!

Route: x.x.x.x
AS_PATH: AS4 AS1

ROA {x.x.x.x, AS1}
Key Findings: ROA Validation

ROA Validation Can be Used to:
• filter mistake hijacks;

ROA Validation Can’t be Used to:
• filter route leaks;
• filter malicious hijacks.

In reality:
• Only 10% of prefixes are signed, transit ISPs doesn’t perform origin validations.
• There is progress at IXes!

Source: https://ripe76.ripe.net/presentations/37-ripe76.azimov.pdf
At least you have half of it!
BGPSec

BGPSec Protocol Specification
RFC 8205

RFC 8205: BGPSec Protocol Specification
RFC 8206: BGPSec Considerations for Autonomous System (AS) Migration
RFC 8207: BGPSec Operational Considerations
RFC 8208: BGPSec Algorithms, Key Formats, and Signature Formats
RFC 8209: A Profile for BGPSec Router Certificates, Certificate Revocation Lists, and Certification Requests
RFC 8210: The Resource Public Key Infrastructure (RPKI) to Router
RFC 8211: Adverse Actions by a Certification Authority (CA) or Repository Manager in the Resource Public Key Infrastructure (RPKI)
AS_PATH Validation

**ASN1**
- pCount
- Flags
- Target ASN2
- Signature

**ASN2**
- pCount
- Flags
- Target ASN3
- Signature

**ASN3**
- pCount
- Flags
- Target ASN4
- Signature
AS_PATH Validation

User Wins!

Route: x.x.x.x
AS_PATH: AS4

ROA \{x.x.x.x, AS1\}
(AS1, AS2) – signed
(AS2, AS3) – signed
(AS4, AS3) – signed
AS_PATH Validation

User Wins!

Route: x.x.x.x
AS_PATH: AS4 AS1

ROA {x.x.x.x, AS1} (AS1, AS2) – signed
(AS2, AS3) – signed
(AS4, AS3) – signed
(AS1, AS4) – illegal
AS_PATH Validation: Bypassed

Attacker Wins!

ROA \{x.x.x.x, AS1\}
(AS1, AS2) – signed
(AS2, AS3) – signed
(AS4, AS3) – not signed
(AS1, AS4) – not signed

Route: x.x.x.x
AS_PATH: AS4 AS1
Key Findings: BGPSec

BGPSec can be used to:

• to detect malicious hijacks at high adoption rate!

In reality:

• Great computation cost;
• Vulnerable for downgrade attacks;
• Nobody is going to use BGPSec!
BGPSec: Unclear who is the rider
Before RPKI
Before BGPSec
There was soBGP
soBGP: Adjacencies

- ISP X publishes information about its connections;
- ISP Y publishes information about its connections;

If there are both pairs \((X,Y) \&\& (Y,X)\) – the pair becomes **trustable**!

If there is only one pair \((X,Y) \| \| (Y, X)\) the pair becomes... **less trustable**!
soBGP: Security Preference

- The pair is trustable: +A
- The pair is less trustable: -B

The route #1 has security preference: 2A – 3B
The route #2 has security preference: A – B
The route #3 has security preference: -2B

Which one is valid and which one is invalid?
soBGP: IXes

AS3 isn’t present in the AS_PATH

No adjacencies between AS1, AS2. Reject?!
Key Findings: soBGP

soBGP Can be Used to:
• Filter **bogon routes**;
• Create security metrics for routes;

soBGP Can’t be Used to:
• **filter route leaks**;

In reality:
• **Problems** with IXes;
• **It’s a rating function**, not a solution.
so-make-bgp-security-by-yourself
<table>
<thead>
<tr>
<th></th>
<th>BGP Hijacks</th>
<th>BGP Route Leaks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mistake</td>
<td>IRR Filters; ROA;</td>
<td>IRR Filters;</td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="#">Route Leak Detection Draft</a></td>
</tr>
<tr>
<td></td>
<td></td>
<td><a href="#">Route Leak Mitigation Draft</a></td>
</tr>
<tr>
<td>Malicious</td>
<td>❌</td>
<td>❌</td>
</tr>
</tbody>
</table>
Are We Doomed for This?
Re-inventing Goals

• Stop propagation of (malicious) hijacks;
• Stop propagation of (malicious) route leaks;
• Incremental deployment;
• Lightweight – no significant changes in BGP!
• Automatically!
Anomaly Propagation
Anomaly Propagation

If we can stop propagation at the level of c2p and p2p – we are done!
A Beautiful Note

If valid route is received from customer or peer it MUST have only customer-to-provider pairs in its AS_PATH.

Then if we have a validated database of customer-to-provider pairs we will be able to verify routes received from customers and providers!
Autonomous System Provider Authorization

ASPA := {
    customer_asn – signer
    provider_asn – authorized to send routes to upper providers or peers
    AFI – IPv4 or IPv6
}

Pair Verification (AS1, AS2)

1. Retrieve all cryptographically valid ASPAs in a selected AFI with a customer value of AS1. This selection forms the set of candidate ASPAs.
2. If the set of candidate ASPAs is empty, then the procedure exits with an outcome of unknown.
3. If there is at least one candidate ASPA where the provider field is AS2, then the procedure exits with an outcome of valid.
4. Otherwise, the procedure exits with an outcome of invalid.

AS_PATH Verification

1. If the closest AS in the AS_PATH is not the receiver's neighbor ASN then procedure halts with the outcome "invalid";
2. If in one of AS_SEQ segments there is a pair (AS(I-1), AS(I)) is "invalid" then the procedure also halts with the outcome "invalid";

ROA \{x.x.x.x, AS1\}
ASPA \{AS1, AS2\}
ASPA \{AS2, AS3\}
ASPA \{AS3, 0\}

Route: x.x.x.x
AS_PATH: AS4
AS_PATH Verification

1. If the closest AS in the AS_PATH is not the receiver's neighbor ASN then procedure halts with the outcome "invalid";
2. If in one of AS_SEQ segments there is a pair (AS(I-1), AS(I)) is "invalid" then the procedure also halts with the outcome "invalid";

Route: x.x.x.x
AS_PATH: AS4 AS1

ROA {x.x.x.x, AS1}
ASPA {AS1, AS2}
ASPA {AS2, AS3}
ASPA {AS3, 0}
AS_PATH Verification

1. If the closest AS in the AS_PATH is not the receiver's neighbor ASN then procedure halts with the outcome "invalid";
2. If in one of AS_SEQ segments there is a pair \((AS(I-1), AS(I))\) is "invalid" then the procedure also halts with the outcome "invalid";

Route: x.x.x.x
AS_PATH: AS4 AS2 AS1

ROA \{x.x.x.x, AS1\}
ASPA \{AS1, AS2\}
ASPA \{AS2, AS3\}
ASPA \{AS3, 0\}
AS_PATH Verification

1. If the closest AS in the AS_PATH is not the receiver's neighbor ASN then procedure halts with the outcome "invalid";
2. If in one of AS_SEQ segments there is a pair (AS(I-1), AS(I)) is "invalid" then the procedure also halts with the outcome "invalid";

Route: x.x.x.x
AS_PATH: **AS2 AS1**
User always wins!
Summary

- ASPA – it’s simple, it scales;
- Works for both route leaks and hijack detection;
- Low computational cost;
- Doesn’t change the protocol itself;
- Works on existing RPKI infrastructure;
- Brings benefit at state of partial adoption.

Limitation: ASN can be attacked by its upstream provider
## BGP Quadrant: Possible Future

<table>
<thead>
<tr>
<th></th>
<th>BGP Hijacks</th>
<th>BGP Route Leaks</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Mistake</strong></td>
<td>ROA</td>
<td>ASPA</td>
</tr>
<tr>
<td><strong>Malicious</strong></td>
<td>ROA + ASPA</td>
<td>ROA + ASPA</td>
</tr>
</tbody>
</table>
Internet Drafts: Published & Adopted

AS_PATH verification procedure:
draft-ietf-sidrops-aspa-verification

ASPA profile:
draft-ietf-sidrops-aspa-profile
The Orchestra

- Alexander Azimov  mitradir@yandex-team.ru
- Eugene Bogomazov  eb@qrator.net
- Eugene Uskov  euskov.cmc@gmail.com
- Randy Bush  randy@psg.com
- Job Snijders  job@ntt.net
- Keyur Patel  keyur@arrcus.com
- Russ Housley  housley@vigilsec.com
BGP Security: Joint Effort

Want to get rid off BGP hijacks/leaks?

• Sign ROAs!
• Try/apply ROV procedure!
• Support ASPA at IETF mailing list;
• Support ASPA as RIR members!
• Make BGP great again!
Let’s make ASPA ASAP!