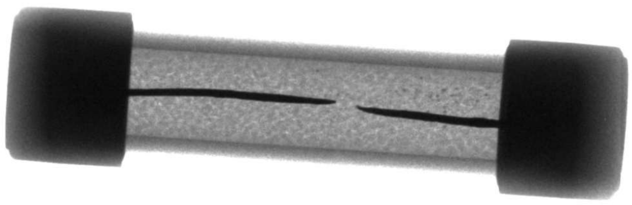
Aug 29 2023

Grave flaws in BGP Error handling



Border Gateway Protocol is the de facto protocol that directs routing decisions between different ISP networks, and is generally known as the "glue" that holds the internet together. It's safe to say that the internet we currently know would not function without working BGP implementations.

However, the software on those networks' routers (I will refer to these as edge devices from now on) that implements BGP has not had a flawless track record. Flaws and problems do exist in commercial and open source implementations of the world's most critical routing protocol.

Most of these flaws are of course benign in the grand scheme of things; they will be issues around things like route filtering, or insertion, or handling withdraws. However a much more scary issue is a BGP bug that can propagate after causing bad behaviour, akin to a computer worm.

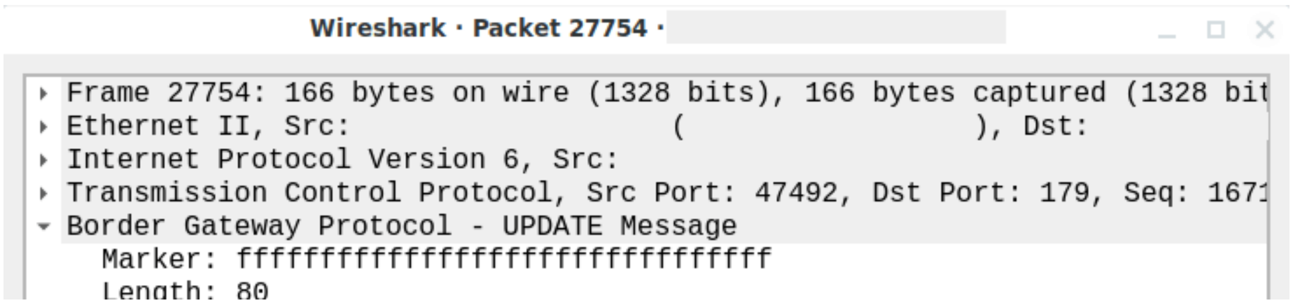
While debugging support for a future feature for my business (bgp.tools) I took a brief diversion to investigate something, and what I came out with might be one of the most concerning things I’ve discovered for the reliability of the internet. To understand the problems, though, we will need a bit more context.

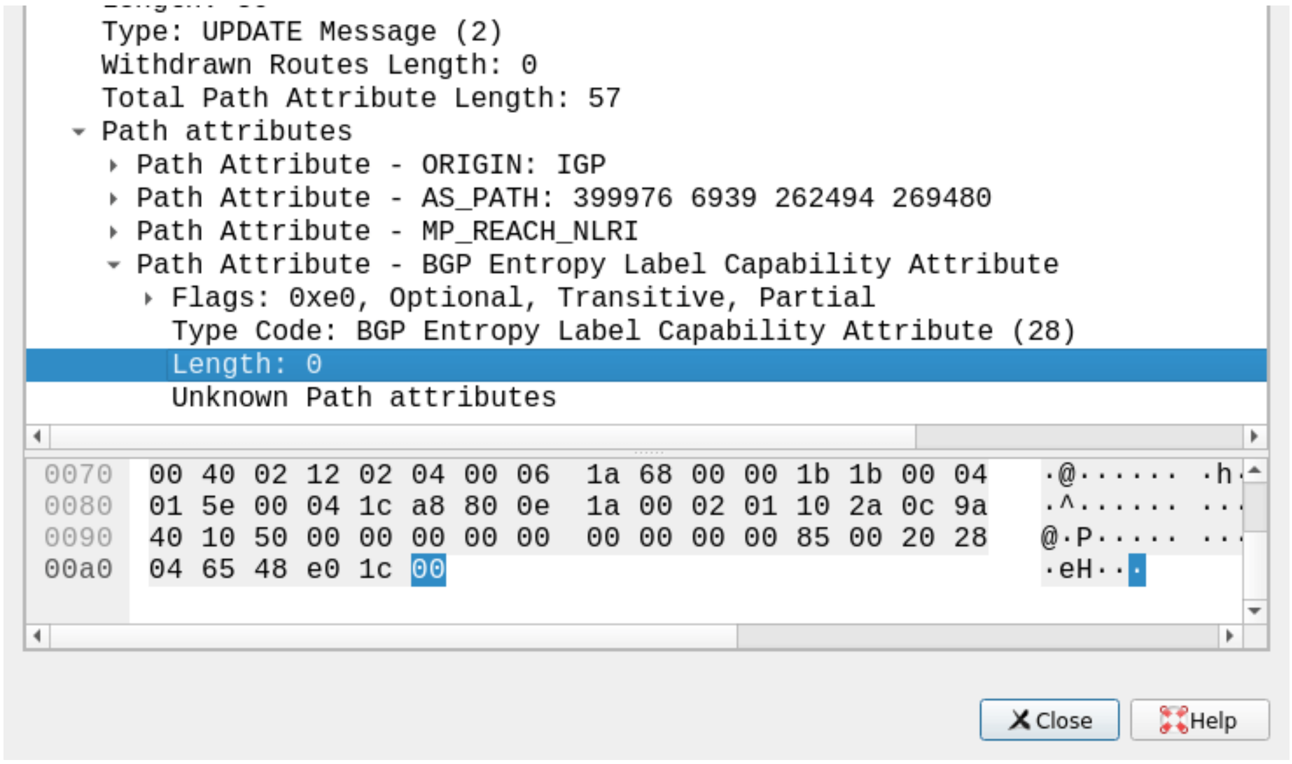
A mistaken attribute

On 2 June 2023, a small Brazilian network (re)announced one of their internet routes with a small bit of information called an attribute that was corrupted. The information on this route was for a feature that had not finished standardisation, but was set up in such a way that if an intermediate router did not understand it, then the intermediate router would pass it on unchanged.

As many routers did not understand this attribute, this was no problem for them. They just took the information and propagated it along. However it turned out that Juniper routers running even slightly modern software did understand this attribute, and since the attribute was corrupted the software in its default configuration would respond by raising an error that would shut down the whole BGP session. Since a BGP session is often a critical part of being “connected” to the wider internet, this resulted in the small Brazilian network disrupting other networks’ ability to communicate with the rest of the internet, despite being 1000’s of miles away.

The packet that causes session shutdowns was really quite benign at first glance:


Wireshark · Packet 27754 ·
```
▸ Frame 27754: 166 bytes on wire (1328 bits), 166 bytes captured (1328 bit
▸ Ethernet II, Src:                        (                    ), Dst:
▸ Internet Protocol Version 6, Src:
▸ Transmission Control Protocol, Src Port: 47492, Dst Port: 179, Seq: 1671
▾ Border Gateway Protocol - UPDATE Message
    Marker: ffffffffffffffffffffffffffffffff
    Length: 80
```



```
    Type: UPDATE Message (2)
    Withdrawn Routes Length: 0
    Total Path Attribute Length: 57
  ▾ Path attributes
    ▸ Path Attribute - ORIGIN: IGP
    ▸ Path Attribute - AS_PATH: 399976 6939 262494 269480
    ▸ Path Attribute - MP_REACH_NLRI
    ▾ Path Attribute - BGP Entropy Label Capability Attribute
      ▸ Flags: 0xe0, Optional, Transitive, Partial
        Type Code: BGP Entropy Label Capability Attribute (28)
        Length: 0
        Unknown Path attributes

0070  00 40 02 12 02 04 00 06  1a 68 00 00 1b 1b 00 04   ·@······ ·h·
0080  01 5e 00 04 1c a8 80 0e  1a 00 02 01 10 2a 0c 9a   ·^······ ···
0090  40 10 50 00 00 00 00 00  00 00 00 00 85 00 20 28   @·P····· ···
00a0  04 65 48 e0 1c 00                                  ·eH···
```

When a BGP session shuts down due to errors, customer network traffic generally stops flowing down that cable until the BGP connection is automatically restarted (typically within seconds to minutes).

This appears to be what happened to a number of different carriers, for example COLT was heavily impacted by this. Their outage is what originally drew some of my attention to this subject area.

To understand why this sort of thing can happen, we'll need to take a deeper look at what BGP route attributes are, and what they're used for.

What is a BGP Route Attribute?

At their core a BGP UPDATEs purpose is to tell another router about some traffic that it can (or can no longer) send to it. However just knowing directly what you can send to another router is not very useful without *context*.

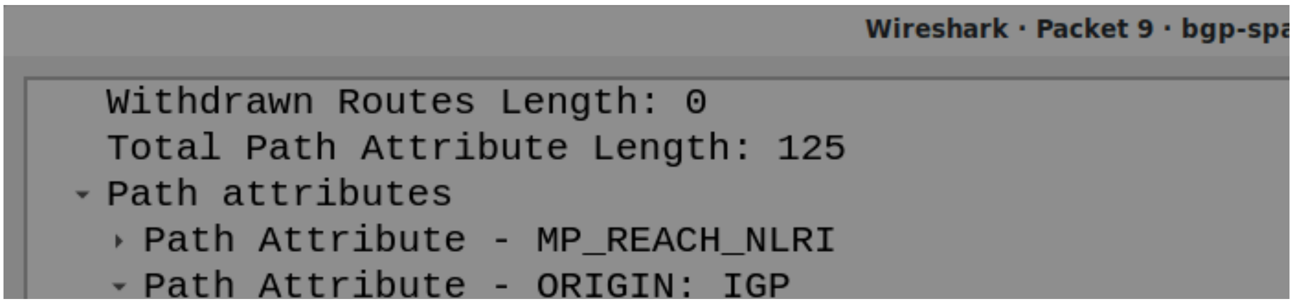
For this reason a BGP packet is split up into two sections: the Network Layer Reachability Information (NLRI) data (aka, the IP address ranges), and the attributes that help describe extra context about that reachability data.

Arguably the most used attribute is the AS_PATH (or actually, the AS4_PATH), an attribute that tells you which networks a route has travelled through to get to you. Routers use this list of networks to pick paths for their traffic that are either the fastest, economically viable, or least congested, playing a critical role in ensuring that things run smoothly.

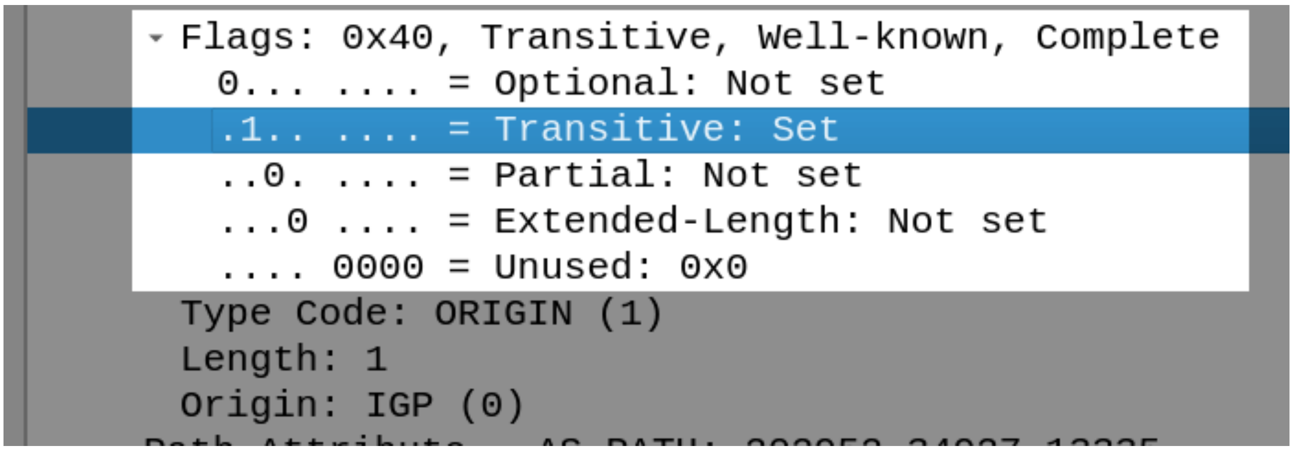
At the time of writing there are over 32 different route attribute types, 14 deprecated ones, and 209 officially unassigned ones. The Internet Assigned Numbers Authority (IANA) is in charge of assigning codes to each BGP attribute type codes, normally off the back of IETF Internet-Drafts. The IANA list doesn't always give the full story, though, as not all internet-drafts make their way into more official documents (like RFC's), so code numbers are assigned (or sometimes even "squatted") to attribute types that did not get wide deployment.

Unknown attribute propagation

At the start of every route attribute is a set of flags, conveying information about the attribute. One important flag is called the "transitive bit":



```
  Withdrawn Routes Length: 0
  Total Path Attribute Length: 125
▾ Path attributes
  ▸ Path Attribute - MP_REACH_NLRI
  ▾ Path Attribute - ORIGIN: IGP
```



```
▾ Flags: 0x40, Transitive, Well-known, Complete
    0... .... = Optional: Not set
    .1.. .... = Transitive: Set
    ..0. .... = Partial: Not set
    ...0 .... = Extended-Length: Not set
    .... 0000 = Unused: 0x0
  Type Code: ORIGIN (1)
  Length: 1
  Origin: IGP (0)
```

If a BGP implementation does not understand an attribute, and the transitive bit is set, it will copy it to another router. If the router does understand the attribute then it may apply its own policy.

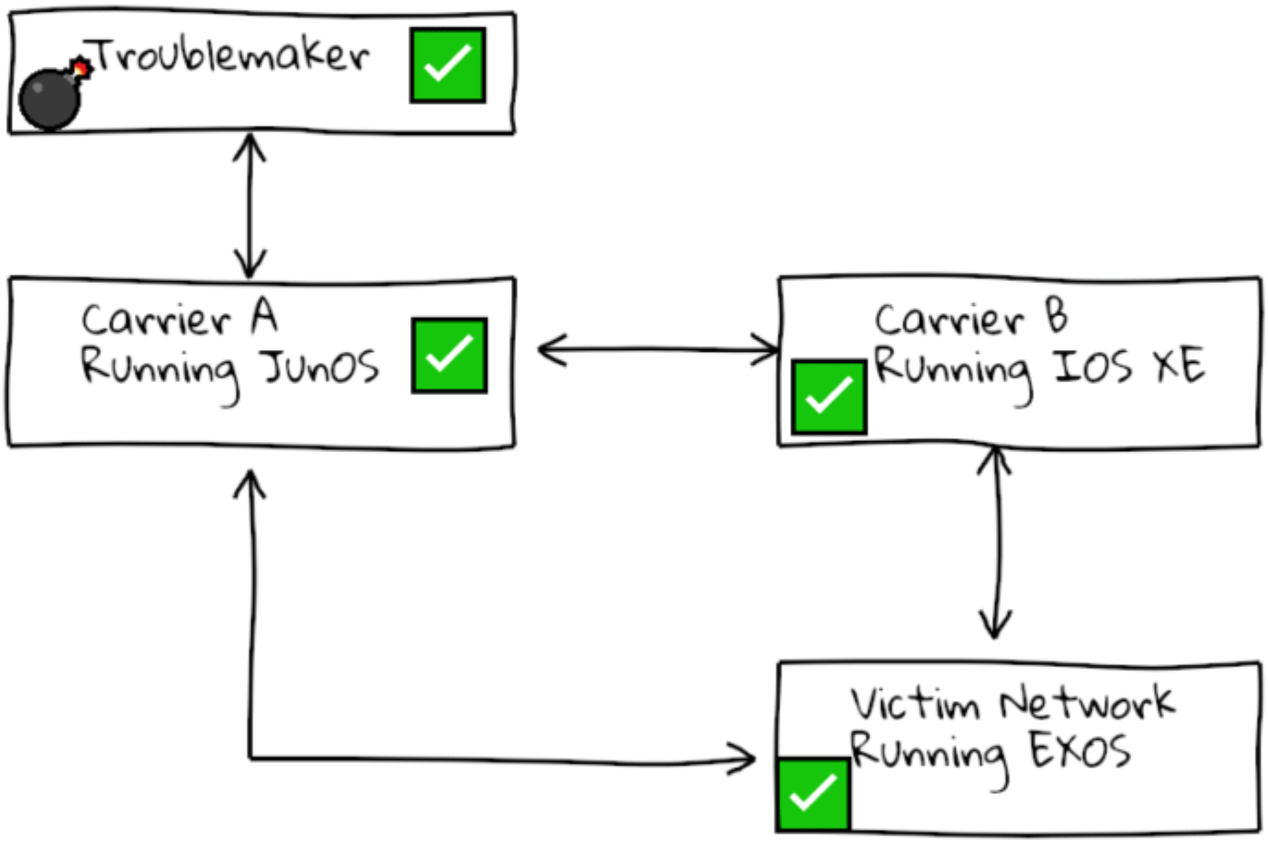
At a glance this “feature” seems like an incredibly bad idea, as it allows possibly unknown information to propagate blindly through systems that do not understand the impact of what they are forwarding. However this feature has also allowed widespread deployment of things like Large Communities to happen faster, and has arguably made deployment of new BGP features possible at all.

When attribute decoding goes wrong

What happens when an attribute fails to decode? The answer depends strongly on if the BGP implementation has been updated to use RFC 7606 logic or not; If the session is *not* RFC 7606 compliant, then typically an error is raised and the session is shut down. If it is, the session can usually continue as normal (except the routes impacted by the decoding error are treated as unreachable).

BGP session shutdowns are particularly undesirable, as they will impact traffic flow along a path. However in the case of a "Transitive" error they can become worm-like. Since not all BGP implementations support the same attributes, an attribute that is unknown to one implementation (and subsequently forwarded along) can cause another implementation to shut down the session it received it from.

With some reasonably educated crafting of a payload, someone could design a BGP UPDATE that "travels" along the internet unharmed, until it reaches a targeted vendor and results in that vendor resetting sessions. If that data comes down the BGP connections that are providing wider internet access for the network, this could result in a network being pulled offline from the internet.



This attack is not even a one-off "hit-and-run", as the "bad" route is still stored in the peer router; when the session restarts the victim router will reset again the moment the route with the crafted payload is transmitted again. This has the potential to cause prolonged internet or peering outages.

This is a large part of why the RFC mentioned earlier, RFC 7606, exists; looking at its security considerations section, we can see a description of this exact problem::

> Security Considerations This specification addresses the vulnerability of a BGP speaker to a potential attack whereby a distant attacker can generate a malformed optional transitive attribute that is not recognized by intervening routers. Since the intervening routers do not recognize the attribute, they propagate it without checking it. When the malformed attribute arrives at a router that does recognize the given attribute type, that router resets the session over which it arrived. Since significant fan-out can occur between the attacker and the routers that do recognize the attribute type, this attack could potentially be particularly harmful.

In a basic BGP setup this is bad, but with extra engineering it could be used to partition large sections of the internet. If BGP sessions between carriers are forced to reset in this way, causing traffic flow to stop, some routes on the internet would not have alternatives to use, making this a family of bugs that is a grave threat to the overall reliability of the internet.

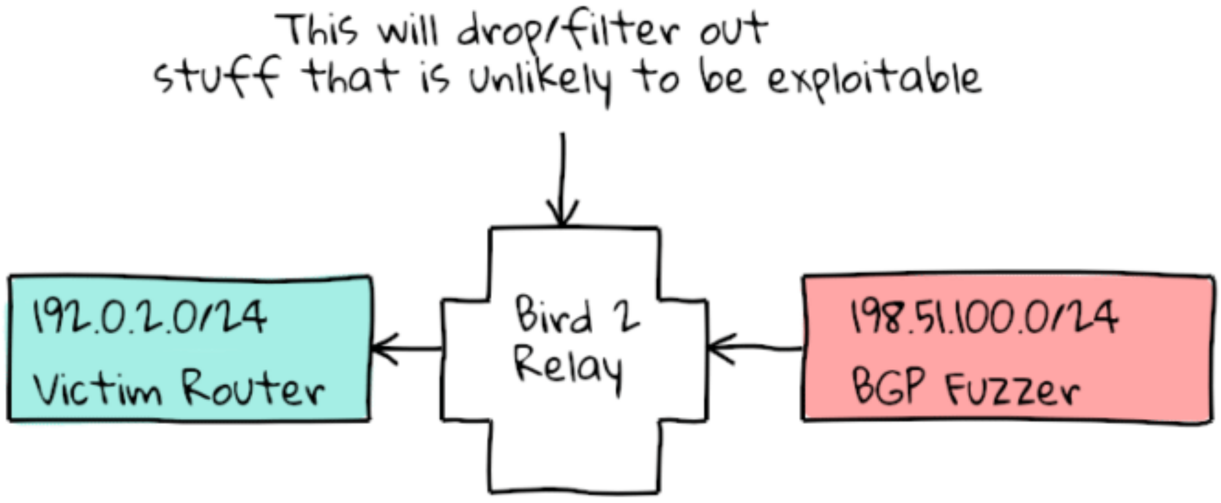
Building a basic fuzzer

Important Commitment: I run a business that involves being peered to many IXP route servers and other peoples routers. I have not and will not ever test for BGP bugs/exploits on customer/partner sessions (unless they give consent).

All testing here has been done either on GNS3 VMs, or physical hardware I have hanging around and in isolated VLANs.

To figure out if this would be a practically exploitable attack, I decided to write a fuzzer that would try to stuff random data in random attribute codes to see if I could get sessions to reset on different vendors BGP implementations.

Because I am looking for problems that are “wormable”, I added a Bird 2 router in between my fuzzer and the router being tested. This way Bird will filter out all of the obvious non-exploitable issues, and leave me with the packets that are of concern.


This will drop/filter out stuff that is unlikely to be exploitable
192.0.2.0/24
Victim Router
Bird 2
Relay
198.51.100.0/24
BGP Fuzzer

All good fuzzers should be able to run unattended, so how do we teach the fuzzer to tell if the session has reset itself? The solution I came to was that the Victim router would always announce a keepalive prefix, 192.0.2.0/24 (aka TEST-NET-1) and the fuzzer would treat a withdraw of that prefix (from the intermediate bird router) as a sign the session went down, and report back the parameters that caused that to happen!

Testing the fuzzer, I can see that the bird output shows unknown attributes as their type code, and a hex encoding of their contents. In addition it puts a `[t]` to indicate that it is transitive.

```
198.51.100.0/24      unicast [fuzzer 21:31:24.378] * (100) [AS65001?]
        via 192.168.5.1 on ens5
        Type: BGP univ
        BGP.origin: Incomplete
        BGP.as_path: 65001
        BGP.next_hop: 192.168.5.1
        BGP.local_pref: 100
        BGP.community: (123,2345)
        BGP.ec [t]: 7d cc c7 30
```

Now that the fuzzer was able to run itself, all that was left was to test all of the vendors one by one…

Fuzzer findings / Impacted Vendors

Keep in mind that the described issues are applicable if you are running an edge device with full BGP tables. If you are not running a “full routing table” or a partial peering table, then you are less likely to be impacted by these discoveries.

Another thing to keep in mind, the issues below are different to the ones that the team at Forescout recently presented at BlackHat.

Unimpacted Vendors:

- MikroTik RouterOS 7+

- Ubiquiti EdgeOS
- Arista EOS
- Huawei NE40
- Cisco IOS-XE / “Classic” / XR
- Bird 1.6, All versions of Bird 2.0

Juniper JunOS Impact

Similar to the problem that caused this research to be done, another exploitable Attribute was found in the form of Attribute 29 (BGP-LS). Due to the nature of the attribute it is unlikely that an exploit attempt will propagate too far over the internet, however peering sessions and route servers are still at risk.

All Juniper users are **urged** to enable `bgp-error-tolerance`:

```
[edit protocols bgp]
root# show
group TRANSIT {
    import import-pol;
    export send-direct;
    peer-as 4200000001;
    local-as 4200000002;
    neighbor 192.0.2.2;
}
bgp-error-tolerance;
```

In all tested cases, enabling bgp-error-tolerance does not reset sessions, and applies the improved behaviour without restarting sessions.

A JunOS software release is expected in the future to correct this. One member of staff at Juniper has also authored an Internet-Draft at the IETF around handling these issues. Juniper is tracking this issue as CVE-2023-4481.

Nokia SR-OS Impact

Fuzzing SR-OS (Version 22.10) revealed many, likely highly propagatable and thus exploitable attributes.

All Nokia SR-OS and SR-Linux users are **urged** to enable `error-handling update-fault-tolerance` on their devices.

```
bgp
    group "TRANSIT"
        export "yes"
        error-handling
            update-fault-tolerance
        exit
        neighbor 192.0.2.2
            peer-as 2
        exit
    exit
    no shutdown
exit
```

In all tested cases, enabling update-fault-tolerance does not reset sessions, and applies the improved behaviour without restarting sessions.

To the best of my understanding, Nokia has no plans to correct these issues, instead suggesting customers apply `error-handling update-fault-tolerance` to their BGP groups.

FRR Impact (and other downstream vendors)

FRR attempts to handle bad attributes using RFC 7606 behaviour. However the fuzzer discovered that a corrupted attribute 23 (Tunnel Encapsulation) will cause a session to go down regardless.

After reporting this bug to FRR maintainers I received an acknowledgement of the issue and understanding that the issue is a DoS risk to FRR users, but I have not managed to get anything out of FRR since.

This bug is being tracked as CVE-2023-38802 and at the time of writing has no patch or fix.

FRR is packaged inside many other products, to name a few: SONIC, PICA8, Cumulus, and DANOS.

OpenBSD OpenBGPd Impact

OpenBGPd also supports the improved RFC 7606 behaviour, however it was found that the recently added Only To Customer implementation could cause session resets. This issue was very rapidly fixed after being reported to them, and is tracked as CVE-2023-38283.

OpenBSD users can install Errata 006 to mitigate this issue.

Extreme Networks EXOS Impact

As a result of fuzzing EXOS, the program revealed 2 highly propagatable and thus exploitable attributes in the form of:

- Attribute 21: AS_PATHLIMIT
- Attribute 25: IPv6 Address Specific Extended Community

There is currently no known patch or mitigating config for this issue.

I made Extreme aware of this problem, however after a back and forth with them waiting for what I understood was an implied release of a patch or fix, they communicated that they will not be fixing it in the near future.

A quote from the security email thread (I've added emphasis to the critical parts of their response):

> After review of all the material, **we are not considering this a vulnerability due to the presence of RFC 7606**, as well as a history of documentation expressing these concerns all the way back to early 2000s, if not earlier. Malformed attributes are not a novel concept as an attack vector to BGP networks, as evidenced by RFC 7606, which is almost a decade old. As such, customers that have chosen to not require or implement RFC 7606 have done so willingly and with knowledge of what is needed to defend against these types of attacks. Thus, the expectation that we'll reset our BGP sessions based on RFC 4271 attribute handling is proper. We do abide by other RFCs, in which we claim support, that update RFC 4271. Other vendors do claim RFC 7606 support and have been sharing these controls as a mitigation to malformed attribute response. They don't appear to be producing new work product to account for these behaviors. **We are evaluating support for RFC 7606 as a future feature.** Obviously, if customers desire a different response, we'll work through our normal feature request pipelines to address. This is no different than any other RFC support request.

I cannot overstate how much I disagree with Extreme's response to this, and in the interests of full transparency (and to avoid any allegations of editorialising this, in my view, extremely poor response), I have made the full email exchange available here: (PDF).

The Vendor Security Response

I’ve been through my fair share of security issue discoveries, and over time I’ve taken a stronger leaning into “simply do not report” or “full disclosure without warning”, rather than the now commonly accepted 90 day “responsible disclosure” methodology. This is mostly because I’ve had really poor experiences when disclosing issues to security teams.

The bugs discussed in this post cover many vendors and implementations. Full disclosure was originally my plan, however due to the clear risk of harm to the general internet routing system from these findings, I felt it was likely inexcusable to do full-disclosure. (Plus, a malicious deployment of these findings could have a small but I believe very real chance of a “kinetic response” from a misunderstanding.)

Overall the response from vendors has been mostly disappointing. One vendor was extremely hard to find contacts for, and I feel that they were stringing me along for some time, only to reply back that they were not going to fix the problem.

Other vendors notably were not immediately interested in notifying customers of mitigating config to the problems, however when I personally started extending notices to my peers at larger carriers about the problems (since if they were not going to, I was going to try and reduce the exploit surface) a vendor notice was issued.

No vendor that I reported to has any form of bug bounty, and this entire adventure has consumed huge volumes of my time and mental capacity. In a normal situation this “cost” would have simply been eaten by an employer whose interest is name recognition or altruistic actions. However I am self-employed with my own company (that admittedly does have an interest in a functioning BGP ecosystem), and so this entire adventure has simply delayed product development that I (and customers) really would have liked to have done.

With all of that in mind, my “good faith” advice to people reporting security bugs in network vendor software is that contacting vendors is not an effective solution.

The people on the other side are either already overloaded, or generally don’t care about receiving issues. Since there is no motivation to report bugs (either in the form of bug bounties, or being treated well), I see no reason to do responsible disclosure (apart from cases where it would clearly protect their innocent customers from misery, while accounting for the risk of a vendor simply doing nothing).

The two options, as far as I see it, are either being strung along by a vendor over email for 90 days and then likely finding out nothing is done, or full disclosure.

I worry about the state of networking vendors.

With that being said, I would like to thank the OpenBSD security team, who very rapidly acknowledged my report, and prepared a patch. My only regret with dealing with the OpenBSD team was reporting the issue to them too quickly.

Closing thoughts

As mentioned before, with a few of the vendors (Nokia, Extreme, Juniper) I found myself contacting their own customers myself to warn them to enable mitigating config, as that proved to be a much more effective way at preventing risk than trying to push the vendor itself into action.

As a result at least several incumbent ISPs, 1 large CDN, and 2 “Tier 1” networks have applied configuration to help prevent these issues from impacting them.

If the goal of reporting security flaws is to reduce harm to their customers, I’m not convinced that reporting problems to vendors has enough of an effective impact to be worth doing, vs the loss of personal time and sanity.

Several people I spoke to have been incredibly helpful (some who could not be listed here). I would like to thank the following people for having some role in helping me either discover/disclose these problems, editing this post, or general support for when vendors were being very frustrating.

- Basil Fillan
- Alistair Mackenzie
- Will Hargrave
- Filippo Valsorda
- Joseph Lorenzo Hall
- Job Snijders
- eta

If you want to stay up to date with the blog you can use the RSS feed or you can follow me on Mastodon/Fediverse @benjojo@benjojo.co.uk

If you run a network and are interested in BGP monitoring do check out bgp.tools! Otherwise if you like what I do or think that you could do with some of my bizarre areas of knowledge I am also open for contract work, please contact me over at workwith@benjojo.co.uk!

Until next time!

Related Posts:

Auditing GitHub users' SSH key quality (2015)

Are BGPs security features working yet? (2018)

Random Post:

Giving every Tor Hidden Service a IPv6 address (2018)