BREAKING HTTPS WITH BGP HIJACKING

Artyom Gavrichenkov
R&D Team Lead, Qrator Labs
ag@qrator.net

QRATOR

Abstract

As of 2015, BGP hijacking and route leaking happen quite often. The most outstanding case is Telekom Malaysia leaking about 179,000 prefixes to its upstream Level3, but there are also thousands of smaller-scale route leaks happening every day. Internet routing was designed on the basis of the trusted parties (Internet service providers) conversing with each other: this makes the task of mitigating all consequences of unwanted route leaks really hard to achieve.

However, BGP hijacking is not often considered as a significant risk. It is known to cause a denial of service, or man-in-the middle attacks in unencrypted plaintext connections, but with strong TLS-backed encryption it's highly unlikely an attacker will have an access to any important data, so there seems to be little reason for the attacker to perform the hijacking.

However, the encryption is backed by TLS and SSL/TLS PKI, which itself depends on Internet routing. This dependance might be crucial for the widely adopted end-to-end encryption, because PKI faces the same risks that apply to all other Internet services. The Internet community has yet to discover possible ways to prevent those risks.

Overview of BGP hijacking

BGP hijacking is known to be caused by Internet service providers (ISPs) which do not filter the prefix announcements coming from some of their peers before transferring them to the

others. Once an attacker obtains access to such peer, he can advertise any prefix he wants to the other peers of the vulnerable ISP, causing denial of service (or possibly man-in-the-middle attack) for either the hijacked prefix or the upstream ISP itself in case the incoming traffic requires more bandwidth than this ISP can handle.

However, the concept of BGP hijacking is quite more sophisticated, because filtering advertisements is itself a complex task for a large transit AS. And what makes it even more complex is that the attacker doesn't have to originate the prefix – it is possible to fake the existing AS path and announce the prefix with the origin of its owner. The prefix may as well “leak” from another ISP. The strict definitions, as proposed by IETF draft *(dickson-sidr-route-leak-def)*, are:

> Hijacked Route: A route which has been originated by a party other than the owner of the prefix. This could be via a forged ASN, or from another ASN.
>
> Route Leak: any Route where, somewhere in the Path, a Non-Customer Route was received over a Peer or Customer Link.

The first one is kind of easier to detect and mitigate. For example, RPKI and BGPsec address exactly this issue. However, RPKI itself may be insufficient for the task, and BGPsec is still in draft as of now, and it'll take years for it to be implemented by the broad Internet community. Last but not the least, an ISP should make some efforts to secure its network, and as there's still no procedure to enforce those efforts, it will be quite an optimistic move to rely on all ISPs in the Internet to do so at will.

There are a lot of prefixes originated from multiple autonomous systems (AS) in the Internet right now. Often, multiple originating AS (MOAS) are there for a reason (for instance, the prefix is announced both by its owner and by a CDN), yet it's impossible to ensure that this is done correctly in any case.

The second one – a route leak – is much harder to understand, let alone prevent without a complex network monitoring solution. Studies show that, on average, there are about 5000 prefixes

leaked in the Internet at any time. Most of them, if not all, aren't caused by any intentional activity and are just a result of human missteps such as simple border router misconfiguration.

Some of the leaks are discovered within minutes since their initiation. Such is the recent case when on June, 12, 2015, Telekom Malaysia (AS4788) started to announce about 179,000 prefixes to its upstream Level3 (AS3549), which in turn propagated those prefixes to its customers. After 2-3 minutes, Level3 customers began to route incoming packets for all those prefixes (including large portion of Facebook prefixes) to Telekom Malaysia, and the latter was unable to handle such amount of traffic. The event caused service degradation, Internet slowdown and packet loss in almost all parts of the world, including Europe and US. But due to the nature of the issue, the root cause was clearly seen from packet routes and AS paths, and it was resolved completely within 2 hours.

Most of the leaks stay for minutes or days. But, as we see, some of the leaks come undiscovered even for years, until something finally breaks in transit. There are a lot of reasons why they go undiscovered: often route leaking involves several operators from different countries, and it is impossible to handle the issue from within one of them; the degradation of quality caused by a leak usually affects only few customers, and the ISP decides that the problem is on the customer's side; and so on.

Under these circumstances, hijacking (or “leaking”) of a network prefix is theoretically possible and practically doable.

There are a few ways to detect hijacking and leaking, but they prove to be either hard to implement or insufficient. Those ways include:

- Constantly analysing Routeviews and AS paths in looking glasses of large ISPs. However, this data is of little use without a complex monitoring solution;
- Measuring TTL field of incoming IP packets. However, TTL is

also easy for a man-in-the-middle to forge;
- Measuring round-trip time (RTT).

We are going to discuss both problems – hijacked routes and route leaks – at once, because the consequences for TLS PKI are the same in both cases. Everything said below about route leaking applies to hijacking as well, and vice versa. We will always use the term “hijacking” throughout the paper, as hijacking (unlike route leaking) often implies malicious activity, and we are going to focus on such activity and its consequences.

Global and Local Hijacking

Hijacks and leaks are often considered to be rare and caused by intentional activity: individial hackers, government-employed task forces or so. As we can see, those intentional activities are indeed rare, but route leaks are nevertheless common, caused by human mistakes. The question is: why attackers ignore the possibility of stealing network prefixes?

Large ISPs often implement some measures to prevent hijacking and leaking, however, the recent case with Telekom Malaysia shows that those measures are sometimes insufficient. Small ISPs often do not care about prefix filtering. Moreover, for the same reasons those ISPs often have their network equipment and even the border routers unpatched and vulnerable, so it may seem not a difficult task for a skilled attacker to break into an ISP's network and to tweak announces at will. However, as we see, this seems to never or almost never happen in the wild.

Our assumption is: when speaking about hijacking, we always mean intercepting the whole network prefix from nearly all over the Internet; and intentional hijacking is rare because the attacker often thinks as well about hijacking a network announce *globally*.

Consider the following case:

1. Prefix X.Y.Z.0/**22** belongs to AS A, which announces it to its upstream AS C.
2. One day, AS M announces X.Y.Z.0/**23** to its upstream AS B.
3. The routing in most intermediate devices is designed to prefer more specific route over the less specific one. There are cases when the opposite happens: say, an ISP sets default route to its upstream provider and then imports routes from local Internet exchange. In this case, local routes will be preferred over anything that comes from the Internet. Such cases are often considered a misconfiguration, but actually it's not exactly forbidden to do so. However, such situation is quite rare, and often more specific route /23 will be preferred over /22.
4. All traffic to X.Y.Z.1 starts to flow into AS M via AS B.
5. AS A notices instant traffic drop. All users of X.Y.Z.1 immediately notice increased latency.
6. It may take some minutes or hours (if, say, happening at midnight), but the root cause of the issue is discovered by AS A. During next few business days, AS A tries to get in touch with AS M, then escalates the issue to AS B, and then finally the problem is going to be solved via the administrative path.

Let's call this “global hijacking”.

As we see, the AS M will finally undergo some investigation and is likely to attract public attention. Then, the permissions to set up routes in AS M are going to be restricted, so AS M can be used in such an attack only once. The attack becomes quite expensive. During the investigation, the attacker may even get caught, which makes the stakes go even higher. And the only real impact will be a denial of service, which often can be achieved easier via amplification or application-level DDoS attacks.

Due to the nature of global hijacking, malicious player doesn't find a reason to do it. However, there are ways to hijack a prefix "locally" – in a couple of autonomous systems. Consider the case:

1. Prefix X.Y.Z.0**/22** belongs to AS A, which announces it to its upstream AS C.
2. One day, AS M announces X.Y.Z.0**/22** to its upstream AS B.
3. What happens next depends on the AS path between B and C and the relations between ISPs on that path.

   Let's say B and C are interconnected. The law is: customer routes are preferred over provider's routes. Then:

   a. If B is C's *customer:*
      B will prefer the route originating from M;
      and C will prefer the route originating from A or B.

      If C prefers the route from B, the hijacking may then be *global*.

   b. If B is C's *provider:*
      C will prefer the route originating from A;
      and B will prefer the route originating from C or M.

      If B prefers route from C, the hijacking will not occur; if B prefers route from M, the hijacking will be local to B and will not propagate to C and to the rest of the world, at least via C.

This simple case illustrates an idea: the hijacking may be "local". In real cases, routing policies will also help to do hijacking locally. The ability to hijack a prefix locally and isolate its advertisement to a small geographic region is obvious, as it is, in fact, the thing that makes BGP anycast possible. But it also opens a possibility to hijack a prefix in such way that the hijack will not be seen on looking glasses of large ISPs, and RTT for most customers will not change either. In case of a local hijacking far away from the origin (for example, in a country from another region), the victim will hardly notice any difference.

Hijacking a Certificate Authority

The procedure of obtaining a TLS certificate for a domain from a TLS certificate authority (CA) is generally as follows:

1. An account is created at the Web site of a certificate authority.
2. A certificate signing request (CSR) is created and uploaded. Though it's highly discouraged, some CAs even allow customer to skip this step and take a private key from the CA.
3. CA offers plenty of options to verify domain ownership:
   - Customer may tweak WHOIS records
   - Customer may upload a specific HTML page to be accessible under a specific URL
   - Customer may set up a custom token in DNS TXT record

   and so on.
4. After the ownership is verified, the customer should pay for certificate (or sometimes should not) and then the CA sends him his TLS certificate. The certificate is valid for next few months or years and may be used to prove the identity of customer's Web site to its users.

If we choose a CA properly, we can carry out a BGP hijacking in such way that will interrupt conversation between CA and the victim (or victim's domain registrar) and will go undetected from victim's view: for example, victim's server and customers are in the US and the prefix is hijacked local to China. We then can impersonate the victim itself (as no reliable identification method is used by CA at that stage) or impersonate a WHOIS server of his registrar and obtain a perfectly valid TLS certificate for the target domain. The procedure is going to take 5-10 minutes at most, and after that the attacker may stop announcing the victim's prefix, so the victim will

have only those 5-10 minutes to detect the issue. Though the CA is likely to be far from the victim in the sense of the Internet topology, the certificate will be valid globally, so it may be used in man-in-the-middle attacks anywhere in the world.

This is a case with ordinary X.509 certificates. The guidelines for issuing Extended Validation (EV) certificates are much more strict, however, identifying the domain is an important step in those guidelines as well. Nevertheless, the possibility to break EV guidelines using the same technique lies out of scope of this paper.

And vice versa: just as we can hijack victim's network prefix near the CA, we can as well steal CA's network prefix near the victim. We can call it “certificate authority hijacking”. It may be more appropriate to hijack the CA if the victim is known to monitor its network prefixes and route leaks. The implementation becomes a bit more complicated, to mangle packets on the way back you need a simple DPI instead of just listening server, but it is just as achievable.

To carry out an attack in such way, you need two things:

1. A border router under your control;
2. Information about your BGP peers: their customers, providers, peerings. Public services such as Qrator Radar (http://radar.qrator.net/) or BGPMon (http://bgpmon.net/) figure out this information on an hourly basis, using public data only: AS Paths, traceroute, etc.

Mitigations

At the moment, we have some workaround. There are monitoring systems (Qrator Radar, BGPMon, Dyn) which will notify the victim in case his prefix is being hijacked. An important thing is: certificate authorities should also make use of those monitoring

solutions, as the problem can occur on both ends.

RFC 7469 also addresses the issue, and, speaking about HTTPS, it is a decent concept, but:

- it's a draft now (just like BGPsec);
- it is once again a workaround, moving the focus of the problem away from the CA to its clients;
- the same approach needs to be applied to all other affected protocols (SMTP, IMAP, XMPP and so on).

Another mitigation possibilities include browser plugins (such as Certificate Patrol for Firefox) and so on.

However, to prevent the issue from happening, we have to tweak TLS PKI and/or Internet routing. TLS PKI is hard to change, but Internet routing is much more harder. The new PKI concepts such as DNS-Based Authentication of Named Entities (DANE, RFC 6698) are also a way to mitigate the problem.

Conclusions

There are flaws in Internet routing and in TLS PKI concept. There are also corresponding risks, but these risks can be mitigated. A better PKI design should make this mitigation a lot easier.

The whole issue isn't about a vulnerability in a software, it is about flaws in a concept. The abstraction of trusted Internet routing is wrong, much more wrong than we got used to think, and we need to find out what to do with the concept rather than implementation. This is a discussion topic for the broad Internet community.

Aside from TLS PKI, BGP monitoring systems are useful for instant detection of networking issues. Today only a fraction of information security specialists and network engineers use them, but, as we can see, those tools may indeed be useful.

References

Route Leaks – Definitions. Internet Draft, v03. Brian Dickson, 2012.
https://tools.ietf.org/html/draft-dickson-sidr-route-leak-def-03

Internet Services: Distributed Mechanisms of Degradation. Oleksiy Semenyaka, 2015.
http://www.slideshare.net/alexanderlyamin/20150525-wsis

Massive Route Leak Causes Internet Slowdown. Andree Toonk, 2015.
http://www.bgpmon.net/massive-route-leak-cause-internet-slowdown/

RFC 7469 - Public Key Pinning Extension for HTTP.
https://tools.ietf.org/html/rfc7469

Certificate Patrol :: Add-ons for Firefox.
https://addons.mozilla.org/en-us/firefox/addon/certificate-patrol/

The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA
https://tools.ietf.org/html/rfc6698