Universiteit van Amsterdam
System and Network Engineering

Research Project 1

BGP Hijack Alert System

Jeroen Schutrup
jeroen.schutrup@os3.nl

Bram ter Borch
bram.terborch@os3.nl

7 February 2016

Abstract

The Internet's main routing protocol is called Border Gateway Protocol (BGP). Everyone using the Internet relies on BGP to work properly. Back in 1989 when BGP was developed, it was not developed from a security perspective. Nowadays everyone knows the Internet is a place filled with malicious users trying to gain benefits from breaking services provided on the Internet. BGP is an open door for the malicious users. It's not incredibly difficult to borrow someone else's IP space when having access to a BGP peering. This is also known as a BGP hijack.

BGP hijacks come in different forms, this paper explains five types of hijacks including a type that was never mentioned in any research before. A new algorithm for detecting these five different types is discussed and tested. An architecture for the algorithm is created as a proof of concept and a real world full BGP feed is used as test data. The results are discussed within this paper. The algorithm uses IRR-records to validate the origin of a BGP update message. To be sure this is a valid source for the algorithm a small test was performed on all Dutch prefixes to see if they are covered with at least one IRR registration.

Table of Contents

Chapter 1

Introduction

The backbone of the Internet is being routed by a protocol known as the Border Gateway Protocol (BGP). Without this protocol, networks of various Internet Service Providers (ISPs) and institutions would not be able to communicate to each other in a cost-efficient manner. However, BGP has not been designed with security in mind[1]. The lack of security makes it possible to perform BGP hijacks. A BGP hijack can be described as advertising Internet Protocol (IP) prefixes or even Autonomous System Numbers (ASNs) to neighboring routers that don't belong to the advertiser.

The detection of hijacked prefixes and AS numbers has been subject to a number of research papers[2][3]. However, proposed methods of early stage detection of BGP hijacks were not found sufficient. On July 25th 2015, the Dutch newspaper *"De Volkskrant"* reported a hijack of an IP prefix of the Dutch Ministry of Foreign Affairs. Due to these findings, the minister of Foreign Affairs had to answer questions of the Dutch House of Representatives[4].

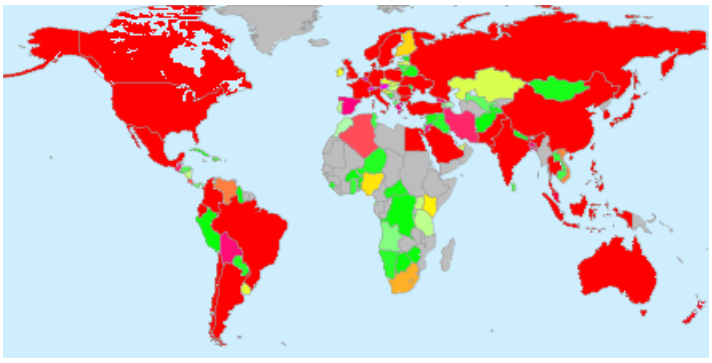

Figure 1.1: Result of the China Telecommunications hijack in 2010. Countries in red have more than 200 prefixes impacted.[5]

In 2010 a China Telecommunications Corporation started to announce about 37000 unique prefixes by mistake[5]. Figure 1.1 illustrates the significant portion of the world that was affected by this mistake. Another example of a BGP hijack originates from 2008, where Pakistani Telecom accidentaly announced a subnet of a prefix owned by YouTube. Due to the nature of BGP, this invalid route was spread over the global Internet, causing availability issues for YouTube all over the world[6].

1.1 Scope

This project aims to detect BGP hijacks near real-time, with a focus on Dutch IP space. Thereby solely using public available information without disclosing any prefix information to third parties. Some papers talk about BGP leaks[2], this means that routes from clients are send to peer routers, so they will advertise these routes to the uplink routers. This will create a false positive because the AS-path is different and for some ASNs not the shortest. This could be detected as a hijack, but it is desired behavior. Detecting BGP leaks is out of scope for this project.

1.2 Layout

The layout of this paper is as follows. Existing work will be discussed first, whereafter the problem will be discussed in chapter 3. After a research question has been defined, a new model to detect BGP hijacks is described in chapter 4. Next, experimentations are discussed and their results reviewed. This report will conclude with a wrap-up of the results, whereafter there'll be a final discussion on future work.

1.3 Terminology

While reading this paper some terms will occur frequently. It is important to know what these terms mean in this paper. When the term prefix is mentioned it means an IP network block written in the Classless Inter-Domain Routing (CIDR) notation. i.e. 145.92.0.0/16. A subnet is a small part of a specific prefix referred to in the text. i.e. 145.92.160.64/26 is a subnet from prefix 145.92.0.0/16. The term supernet is used to refer to a larger network than the specific prefix referred to in the text. i.e. 145.80.0.0/12 is a supernet for prefix 145.92.0.0/16. BGP routers containing the full BGP routing table don't need a default route. They know how to route traffic to every network because they have mostly directly connected neighbors they peer with. The 'zone' these routers are in, is called the Default Free Zone (DFZ).

Chapter 2

Related work

This chapter aims to give an introduction into BGP hijacking and discuss efforts that have been put into detection and symptoms of BGP hijacking. The first section gives an overview of different types of BGP hijacks. Following up, current available solutions and methods to detect prefix hijacks are described, and each of their respective limitations will be elaborated on. Finally, the features which are found interesting for this research are explained.

2.1 BGP hijack types

The paper of *Hu et al.*[2] describes five types of BGP hijacks, four of these are considered relevant for this project. These attacks concern the hijacking of a prefix or a subnet of a prefix, advertised by an AS different from the origin AS. Other types of attacks are announcing the prefix or a subnet, but advertised by a third party under the same ASN as the original ASN. The fifth type in the paper of *Hu et al.*,which is out of scope for this project, concerns an event also known as BGP leaking[7].

1. **Hijack a prefix:** A full prefix owned by an organization is advertised by an AS which does not have ownership over this prefix.
2. **Hijack a subnet of a prefix:** A subnet of the prefix from an organization is advertised by an AS which does not have ownership over this prefix.
3. **Hijack a prefix and its AS:** The AS number and its prefix are being advertised by an organization which does not have ownership over these objects.
4. **Hijack a subnet and its AS:** A subnet of the prefix and its original AS number are being advertised by an organization which does not have ownership over these objects.

In case of a prefix hijack, an entire, already announced prefix is advertised once more by another administrative domain. The basics of BGP ensure that only preferred routes are advertised to BGP peers. Upon receiving a more expensive, a.k.a. a less preferred route, a BGP router will save this route into its BGP table to be used as a fallback route. This fallback route will only be advertised to its neighbors when the preferred route is no longer valid. A consequence of this behavior is, since all routers already have a route to the original prefix, that the hijack will only be noticed by a limited number

of nodes in the BGP infrastructure. When observing a subnet hijack, the availability of the subnet is unique and will be propagated throughout the Internet. Therefore, such an event will be witnessed by all BGP nodes, regardless of the originating ASN.

2.2 Available tools and methods

Monitoring the Border Gateway Protocol can be done in different ways. This research divides solutions by the way they gain information to detect a prefix hijack. These approaches are displayed in a comparison chart, shown in Appendix A. This and upcoming section will discuss these methods and their feature sets.

Some tools use control-plane information to detect hijacks, while other solutions detect hijacks by data-plane information. Control-plane information is information from the router itself, like a BGP feed or the BGP table of the router. In contrary, data-plane information comes from information sources effected by control-plane decisions. For example, a routing decision on the control-plane effects a data-plane ICMP traceroute. Existing tools utilize control-plane as well as data-plane information. For example, the approach of *Zheng et al*[3] primarily uses control-plane data. However, in the case of a suspected hijack it uses data-plane information to verify the validity of the hijack.

Web services
Web services like BGPmon[8] and Dyn.com[9] commercialized BGP prefix monitoring. As these web services are closed source they don't offer insight in methods used to detect prefix hijacks. For some organizations it is not desirable to use such webservices because they are limited in the number of prefixes they can monitor and customers need to disclose prefix information.

Theoretical methods
A number of theories regarding BGP hijack alerting have been published[2,3,10,11,12,13,14,15,16]. However, they all come with some limitations, or are not applicable for this project's use case. A method proposed by *Hu et al.*[2] utilizes a full BGP feed to detect anomalies for the monitored prefixes. When an anomaly is detected, the algorithm uses data-plane, e.g. ICMP traceroute information. Moreover, it uses IP packet Identification (ID) values to validate the hijack. Therefore, this system needs live clients within the monitored prefix, as well as clients in network mimicking the original prefix. An approach taken by *Zheng et al.*[3] uses data-plane information to detect a hijack. With this approach, monitoring nodes are strategically placed on the path which is traversed to an Autonomous System. These nodes are effectively functioning as reference points. Assuming that the path from these nodes to the monitored AS should always stay unchanged, hijacks can be detected whenever this path does change. A variety of additional methods[12,13] exist on these kind of data-plane hijack detection schemes. However, utilizing data-plane information is hardly scalable[16], can be countered by the attackers[3] and limits the detection of hijacks on sub-prefixes[3]. Because of the requirements discussed in chapter 3.1, data-plane

methods are not desirable for this project as they often require administrative control over the monitored prefixes, and don't support granular prefix monitoring[16].

iSpy
iSPY[15] is worth mentioning as it, contrary to methods discussed so far, monitors BGP hijacks from the perspective of the prefix itself. By actively probing major external transit networks it effectively tests for Internet connectivity. It thereby distinguishes between regular network failures and prefix hijacks.

Tooling
Theoretical models aside, work has been done on actual implementations[10][17]. The Prefix Hijack Alert System (PHAS) has been one of the first doing so. However, PHAS suffers from a high amount of false positives[3]. This is caused by the fact that it's very hard to distinguish between hijacks and regular changes in the routing topology when using control-plane data[18,16]. PHAS is also fairly late in detecting hijacks, as it comes with a three hour delay[10]. Another available tool is BGPmon.py[17]. This tool, written by *Saif El-Sherei*, relies on a complete baseline completely set by the user, including an origin AS, prefix and its country code. It does however not have a mechanism to automatically update this data, and to detect MOAS conflicts, as the valid upstream provider is not known to the monitoring application.

2.3 Feature comparison

This section helps to further clarify existing methods and their features as displayed in Appendix A. Features in the leftmost column are either referenced from existing papers, or are included because they could be of great value for a hijack detection system, although they were not explicitly mentioned in existing papers. In the next paragraphs, all of these features will be discussed.

Hijack detection types
As discussed, four types of BGP hijacks can be identified[2], and are included as features in the comparison chart. There has not been done a lot of work concerning hijack detection of unused prefixes. This concerns prefixes which are assigned, but should not be announced on the Internet. According to *Vervier et al.*, exactly these prefixes are a popular target among hijackers[18]. In order to lower the amount of false positives, legitimate transfers of prefixes among Autonomous Systems should be detected as well[19].

Multiple Origin AS (MOAS)
Accuracy of control-plane information is degraded when Multiple Origin AS conflicts are observed[3]. It is difficult to distinguish between legitimate MOAS conflicts and prefix hijacks, as in both cases a change of origin AS is observed. MOAS conflicts are usually short-term, and can be caused by multihoming, faulty configurations or the use

of anycast addresses[19]. Identifying MOAS conflicts is key for a hijack detection system, as the number of MOAS conflicts are yearly increasing by roughly 20%[19].

Detection delay & stealthiness
Some existing systems suffer from a high detection delay[16]. As research of RIPElabs shows, it takes approximately 40 seconds for a BGP update to be propagated over the worldwide BGP infrastructure[20]. Therefore, this project intends to realize a real-time hijack detection speed. When monitoring prefixes in another administrative domain, it might not be desirable to probe the monitored network. Existing proposals use port scanning, ICMP requests or TCP handshakes in their hijack verification process[2,15,3]. Although this techniques increase detection accuracy it may not be desirable for an organization to send this kind of traffic to a network which is not under their administrative control. This can occur when monitoring another organization their network. Therefore achieving a stealthy solution is preferrable. Furthermore, these data-plane techniques are only applicable when monitoring a network with at least one online node in it. Even then it is still not able to detect more specific hijacks.

Scalability & information disclosure
Hijack detection schemes based on data-plane information suffer from poor scalability[16]. Therefore, creating an easy scalable anomaly detection scheme is crucial. Although the majority of all reviewed systems don't disclose information into the public, organizations monitoring their prefixes using web services are identifiable when registered to such a service. In order to remain anonymous, preventing information leaking is key to this project, meaning an organization monitoring a prefix should not be traceable for doing so in any way.

Attacker identification
In order to handle and mitigate the effects of BGP hijacks, it is key to get to know information regarding the attacker[15]. Solely using data-plane information won't identify the attacker. Using control-plane data is essential when seeking knowledge regarding a hijacker and his AS or his Internet Service Providers AS (ISP)[16].

Chapter 3

Problem statement

Several existing options and features have been discussed in the previous chapter. According to the comparison table, no ideal solutions exist. Summarizing, some existing methods are web-based, impacting data confidentiality. Second, some methods assume the monitored prefixes fall within the same administrative domain as the team monitoring them. When monitoring machines in another administrative domain, it is not desirable to perform portscans. Furthermore, fingerprinting on the data plane can easily be faked by the attacker. Third, some methods don't allow for near real-time hijack detection. Another limitation concerns unused prefixes, which are not mentioned in most of the papers, although exactly these prefixes are a popular target among hijackers [18].

3.1 Requirements

Restrict data leakage
Preventing information loss to the public domain is of great interest to this project. Although prefix and AS information is already publicly available, it might not be desirable for an organization to disclose information about the monitored prefixes into the public.

Detect hijacks of unused prefixes
Contrary to existing methods, detecting hijacks of unused prefixes should be detected, especially since these kind of attacks gain popularity [18].

Handle legitimate MOAS conflicts
As control-plane BGP detection algorithms have been marked as unreliable [15,16], this project aims to improve on that. Since the use of data-plane information gathering is not desirable for this project, routing information gathering will be limited to control-plane data.

Restrict information sources to public available resources
This paper focuses on monitoring prefixes which are located in a different administrative domain. In order to monitor a prefix, there is no need for any node to be online within this prefix. Furthermore, no administrative control is required over the nodes residing in the monitored prefixes.

BGP hijacks must be detected near real-time
As described by RIPElabs, it takes approximately 40 seconds for a BGP announcement, and about three minutes for a withdrawal to be propagated over the BGP infrastructure[20]. In contrast to the control plane detection system PHAS[10], this project aims to detect BGP hijacks within this timeframe.

3.2 Research question

The requirements mentioned in the previous part can be collaborated into one research question:

How to create an early detection system for BGP hijacks for a fixed number of IP-ranges and AS numbers using public resources?

The research question require three subquestions to be answered before the research question can be answered.

What public resources are available that could be used to detect BGP hijacking, without disclosing IP prefix information?
In order to detect legitimate MOAS conflicts and increase the reliability for control plane data, new information sources need to be leveraged.

What's the reliability of these public resources for monitoring prefix hijacks in The Netherlands?
When detecting hijacks for a large number of prefixes, the obtained data source should be complete so reliable hijack detection can be guaranteed.

How to detect BGP hijacks using this public information, with a low number of false positives?
To create a hijack detection system, an algorithm is needed that leverages aquired information sources.

Chapter 4

Proposed system

This chapter first discusses BGP behavior and its implications on a hijack detection scheme. Including in this section, a new type of hijack is explained. Then, available information sources are discussed, after which the actual detection algorithm will be described. This model will be referred to as the BGP Hijack Alert System (BHAS).

4.1 Design considerations

At this moment (January 26th 2016), the full BGP table counts a total number of 616.022 announced IPv4[21] and IPv6[22] prefixes. Back in 2005, an average of 1600 BGP updates were sent over the Default Free Zone (DFZ) every hour[23]. Routers in the DFZ have no default route, e.g. null route configured[24]. In terms of routing information they are entirely dependent of routes received from their neighbors. Studies predicted a significant growth towards 2.8 million updates per hour at the end of 2010[23].

Figure 4.1 and 4.2 illustrate the continuing trend of the increasing size of the BGP Forward Information Base (FIB) of routers in the DFZ. This table is used by routers when forwarding traffic[24]. Hijack detection systems working with control-plane data need to be able to process this information near real-time.

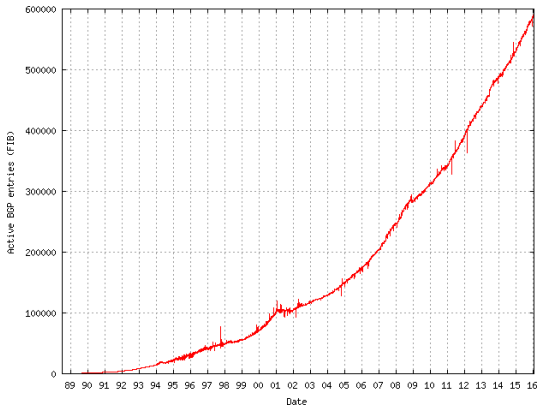

Figure 4.1: The growth of IPv4 prefixes[25]

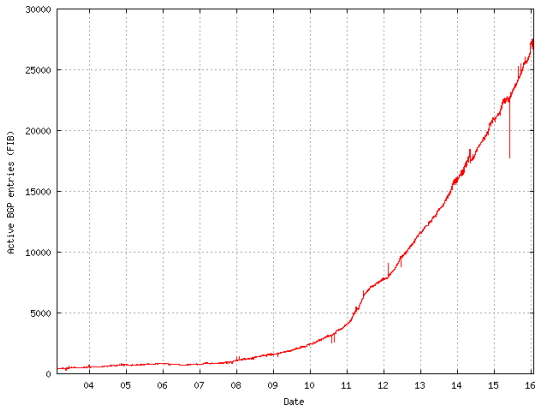

Figure 4.2: The growth of IPv6 prefixes[26]

4.1.1 BGP hijack types

The four types of hijacks specified in section 2.1 will categorize many hijacks. However, another type of hijack was worked out during this research. This is called a less specific, or a supernet hijack (later referenced to as a type 5 hijack). Briefly, a less specific hijack can be described as an announcement of a large IP prefix which itself is not being announced, but has more specific prefixes that are. This leverages the route to spread over the internet. For most network blocks below this route, i.e. networks sharing the network part, but with a higher subnet mask, there will be a more specific route known. These networks won't be affected, as BGP's routing decisions prefer more specific routes[1]. However, depending on the hijacked prefix, there might be an address block which is not yet being explicitly advertised. This IP space within the hijacked prefix will be routed to the malicious router advertising the less specific prefix. Especially unused prefixes are vulnerable for a less specific hijack. The proposed model is capable of detecting all five types of hijacks. In table 4.1 all five hijack types are summarized, including their ID which will be referred to later in this paper.

#	**Name**	**Announcing AS**	**Annnounced Prefix**
1	Prefix hijack[2]	$\neq$ authorized AS	= monitored prefix
2	Subnet hijack[2]	$\neq$ authorized AS	$\subset$ monitored prefix
3	Prefix & AS hijack[2]	= authorized AS	= monitored prefix
4	Subnet & AS hijack[2]	= authorized AS	$\subset$ monitored prefix
5	Supernet hijack	$\neq \vee =$ authorized AS	$\supset$ monitored prefix

Table 4.1: Comparison of all hijack types

4.1.2 Utilized information sources

This section describes all sources of information that will be used for detecting BGP hijacks.

BGP feed

In order to detect prefix hijacks as soon as possible, BGP UPDATE messages are collected from a full BGP feed sent by the core routers of an ISP. A full BGP feed contains a stream of updates from all public AS numbers all over the world. Private AS number updates are not re-advertised by an ISP, these are advertised by an aggregated prefix from the ISP. BGP updates contain, amongst others, an AS path which is displayed in equation figure 4.1. This is the path of Autonomous Systems to traverse from a neighbor peer to the destination AS where the prefix resides. This paper later refers to two elements in the AS path, namely the origin AS and the upstream AS. The origin AS is AS_N, whereas the upstream AS is the second last AS in the AS path, namely (AS_{N-1}).

$$U_{as\_path} = \{AS_0, AS_1, \ldots, AS_{N-1}, AS_N\} \tag{4.1}$$

RIPEstat API
Metadata regarding Autonomous Systems and BGP prefixes are collected through the RIPEstat API[27]. This API (Application Programmable Interface) is maintained by RIPE, the Regional Internet Registry (RIR) of Europe. Its sources of information includes authority data of other RIR's and geographical information of MaxMind[28]. RIPEstat is used to retrieve the correct AS numbers and country codes for every monitored prefix. These AS numbers originate from the Internet Routing Registry (IRR), and represent the Autonomous Systems that are allowed to announce a certain prefix, also known as routing policies[29]. As to be seen in table 4.2, this data shall be of great value when detecting various hijack types. Details regarding the IRR itself will be discussed in the next paragraph.

Resource	**Utilized for**	**Description**
geoloc	BHAS	Determine geolocation of monitored prefix origin AS
geoloc	BHAS	Determine geolocation of BGP update upstream AS
whois	BHAS	IRR records, obtain authorized ASNs for prefix
whois	Research IRR coverage	IRR records, obtain authorized ASNs for prefix
announced-prefixes	Research IRR coverage	Obtaining currently announced prefixes for a given AS

Table 4.2: Utilized resources from RIPEstat API

When querying the RIPEstat API, prefix information is disclosed to RIPE. However, RIPE allows the database to be downloaded. RIPE offers a daily snapshot of the database, in which personal contact information has been omitted, available for download from an FTP mirror[30]. Another option is to anonymize requests through the use of Tor or a Virtual Private Network (VPN).

Internet Routing Registry (IRR)
As mentioned in paragraph 4.1.2, the IRR contains a set of routing policies which is maintained by participating organizations, for example, prefix owners. This model will use the IRR to detect legitimate MOAS conflicts. According to performed research[31,32,29], the IRR is not complete at all. As this research was published in 2009 and earlier, the current coverage of all Dutch ASes in the IRR need to be researched in order to determine whether the IRR is a usable resource. The strategy for doing so will be discussed in the next paragraph.

CIDR report
In order to retrieve a list with all Dutch Autonomous System Numbers, CIDR Report was used. This websites publishes, amongst others a list of issued AS numbers, including per AS country code of the issued party[33].

4.1.3 Excluded information sources

This section explains why certain public resources were not used for this project.

Looking glasses
Looking glasses can provide valuable information. However, it conflicts with the requirement not to leak any private information, like prefixes or AS numbers to third parties. Therefore, the decision was taken not to use them as a source of information for detecting prefix hijacks. However, as almost 200 looking glass server worldwide are open for querying, it is interesting future work to add this information source[34].

RouteViews
The University of Oregon publishes BGP announcement data, collected from several locations over the world. Although this data is approximately two hours behind in time[35], collecting BGP updates from multiple vantage points worldwide can be of great value. These incremental BGP Update messages are MRT[36] formatted and can be collected from an FTP mirror. Since BGP routers only advertise their most preferred route to a destination, a single BGP feed will give a limited view. Utilizing feeds from geographically distant vantage points is advised to improve the near real-time detection of hijacks[37]. As a full BGP feed has already been setup for this project, RouteViews will not be used. For now, it is considered as future work.

4.2 Architecture

This chapter discusses the architecture of the BGP Hijack Alert System. In figure 4.3, the application's architecture can be seen, which will be elaborated on in the upcomping sections. First, the bottom layer will be discussed as this contains elements which will be referred to in the upper layers.

4.2.1 Software router

In order to process the BGP feed a low level entry point into a routers software is required. No vendor will provide this kind of entry point to the source code of a router. Therefore a software router must be used to make this model work. Some different software routers are available. Quagga[38], EXAbgp[39] and BIRD[40] are compared. Quagga is easy to use because of the Cisco like command line interface (cli) but its it harder to create an output for BGP updates. EXAbgp on the other hand is harder to configure and has its own set of commands but the way it outputs received information is very

Feature	**EXAbgp**[39]	**BIRD**[40]	**Quagga**[38]
CLI	None	None	Cisco CLI
Output formats	JSON, plain text	None	MRT
Support program execution	Yes	No	No
Runtime change	No	Yes	Yes
Protocol support	BGP	BGP, RIP, OSPF, ...	BGP, RIP, OSPF, ...

Table 4.3: Software router comparison

easy to use and control. BIRD is the last product compared in table 4.3. BIRD is not able to produce useful output for further analysis. BHAS uses EXAbgp to receive the BGP feed and feeds it to the algorithm.

4.2.2 Storage

BHAS uses Object Relational Mapping (ORM) to map application objects to database entities. Regarding database storage, three tables are used to respectively store the monitored prefixes, their origins and observed hijacks. In figure 4.4, the Entity Relationship Diagram displays these database entities and their properties. A monitored prefix uses a

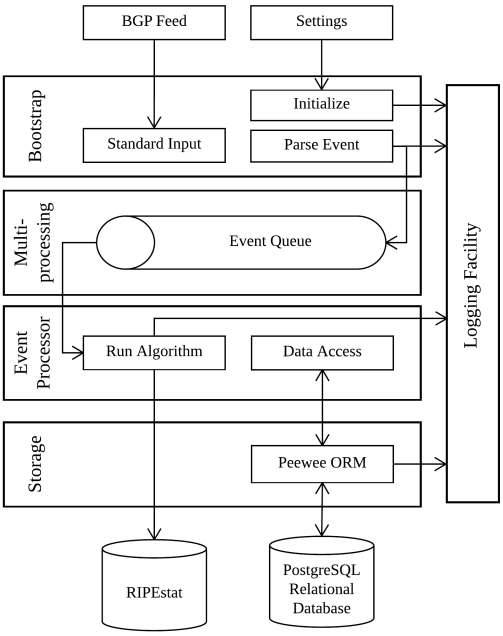

Figure 4.3: BHAS Architecture

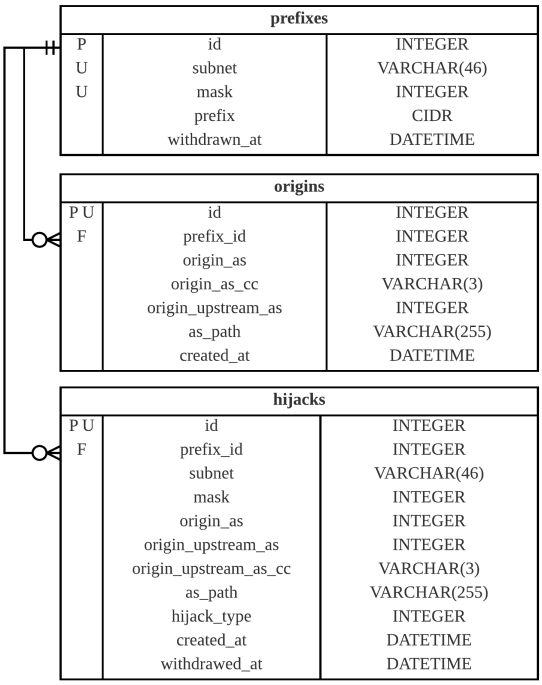

Figure 4.4: BHAS Entity Relationship Diagram (ERD)

CIDR datatype which is only available in PostgreSQL and allows the database engine to compare prefixes[41]. Furthermore, a prefix is related to zero or more origins. An origin represents an Autonomous System authorized to announce a prefix, which must always be related to exactly one prefix. A prefix might have no origin at all, when this prefix should not be announced by any AS at all.

Alike origins, a hijack should also be linked to exactly one prefix. For every hijack, the AS announcing the sub-or-supernet of the monitored prefix are saved, as well as the upstream AS and its country code, and the complete AS path. The hijack type relates to the five types discussed in section 4.1.1.

4.2.3 Initialization

As BHAS only monitors a fixed number of prefixes, its database needs to be initialized before the application can start monitoring. A textfile containing a newline seperated list with prefixes in CIDR notation is expected as input. The system will then query all IRR records for every prefix, as well as every prefix's country code and save the result into the database. Per prefix, a new Origin is created for every AS found in the IRR records. The AS Path will be left blank, as it can only be determined upon receiving a BGP announcement originating from this origin.

4.2.4 Bootstrap

When starting BHAS, the bootstrap component will wait for JSON formatted standard input. The JSON output format of ExaBGP is used as input format for this model. The bootstrapper will try to parse the input into a temporary Event object. It will do so for every prefix encountered in an update message. If successful, the Event is put onto a queue described in the next section. The result of the parsing is written to the logging facility, irrespective of whether the attempt failed or succeeded.

4.2.5 Multiprocessing

The main program of BHAS utilizes two processes, sharing a queue. While one process converts *stdin* to Event objects which it places on the queue, the second process which is described in the next paragraph runs the algorithm for every object on the queue. Since the order of the queue objects matters, the decision was made not to use multiple processes simultaneously processing the queue. Hence, only two processes exists. Still, multiprocessing enables the program to process the input fast, without having to wait for the bottom layers to complete API calls or database access.

4.2.6 Event processor

Together with the Bootstrap, this is BHAS its main component. The event processor pulls Event objects off the queue in a FIFO (First-In First-Out) manner. These objects are then fed to the algorithm, which will be described in section 4.3. At this point, the database may be accessed and HTTP calls are performed to the RIPEstat API.

The result of the algorithm is written to the logging facility in order to ease debugging. Depending on the logging settings, intermediate results may be logged as wel.

Attribute	**Description**
prefix	The announced network noted in CIDR format.
originAS	This is the AS which is announcing the prefix in the received update.
ASPath	The complete AS path included in the update, see equation 4.1.
upstreamAS	This is $ASPath_{N-1}$.

Table 4.4: Event object properties

4.3 Algorithm

In this section, the algorithm function in the Event Processor component is reviewed, see figure 4.3. The BHAS algorithm can be seen as a flowchart in appendix B, and the pseudocode is displayed within this chapter, in algorithm 1.

As explained in the previous section, every prefix in a BGP update is parsed to an Event object. Eventually, this object will be fed to the algorithm. As an Event is a temporary object, it is not mentioned in the ERD, figure 4.4. Therefore, all Event object properties relevant to the algorithm are described in table 4.4.

Upon entering the algorithm, the prefix is tested on checked whether it is a subnet, an exact match or a supernet of a network of interest set of prefixes. This set is denoted as P. When no matches are found, the update will be discarded. When it does match with an entry in P, BHAS checks whether the update is an announcement or a withdrawal.

Announcements
If the update is an announcement, an origin is searched such that the origin is related to p, and the origin's AS matches the announcement's AS. If so, the origin is considered equal.

Whenever no matching origin is found, it might be the case that the monitored prefix has been transferred to another Autonomous System. In order to verify, the prefix its latest IRR records are queried from RIPEstat. If an origin in the IRR records is tested equal to an origin in $p_{origins}$, the origin is added to the database. If not, a hijack alert is raised. Depending on the announced prefix to be a sub-or-superset of the monitored prefix, the hijack will be either of type one, two or five.

If a matching origin, o, was found, the AS path of o is compared to the AS path advertised with the update u. If these values correspond, the announcement is considered to originate from the authorized origin. As announcements are only propagated on receiving a better path to a destination, or when the preferred path has been invalidated, the announcement might be caused by a canceled hijack. Therefore, the checkHijack function is initiated (see paragraph 4.3).

Algorithm 1 Hijack Detection Algorithm

```
1: procedure PROCESS PREFIX UPDATE(update u)
2:     if ∃p ∈ P | (p_prefix ⊇ u_prefix) ∨ (p_prefix ⊆ u_prefix) then
3:         p ← (p ∈ P | (p_prefix ⊇ u_prefix) ∨ (p_prefix ⊆ u_prefix))
4:         if u_type = announcement then
5:             if ∃o ∈ O | (o_AS = u_originAS) ∧ (o_prefix = p) then
6:                 o ← (o ∈ O | (o_AS = u_originAS) ∧ (o_prefix = p))
7:                 if u_ASPath ≠ o_ASPath then
8:                     if u_upstreamAS = p_upstreamAS then
9:                         checkIfHijacked(p)
10:                    else
11:                        g ← getGeolocation(u_upstreamAS)
12:                        if g = o_geolocation then checkIfHijacked(p)
13:                        else hijackAlert(p, u)
14:                else checkIfHijacked(p)
15:            else
16:                R ← getLatestIRR(p)
17:                if ∃r ∈ R | r_originAS = u_originAS then updateDb(p, r)
18:                else hijackAlert(p, u)
19:        else if u_type = withdrawal then
20:            if ∃h ∈ H | (h_prefix = p) ∧ (¬h_withdrawnAt) then clearHijack(p)
21:            else withdrawPrefix(p)
22:    else discard(u)
23: end procedure
24: function CHECKIFHIJACKED(prefix p)
25:    if ∃h ∈ H | (h_prefix = p) ∧ (¬h_withdrawnAt) then clearHijack(p)
26:    else discard(u)
27: end function
```

In case the AS path of the origin o and update u did not correspond, a legitimate AS is announcing a monitored prefix from an upstream provider that has never been observed doing so. This might be a legitimate MOAS conflict, for example, when multihoming, caused by a fallback to a secondary ISP. In this case, BHAS assumes the upstream provider should have at least one prefix registered in the same country as the origin AS originates. To perform this check, geolocation data of $u_{upstreamAS}$ is downloaded from the RIPEstat API. If the geolocation data does not match, an AS hijack alert of type three or four is raised.

Withdrawals

If the update is checked as a subnet, exact match or a supernet and is recognized as a withdrawal, BHAS will check if a hijack h exists in the set of hijacks H where the hijack

is related to p. If so, these sub-or-supernet hijacks will be set as withdrawn. When no hijacks are found, the prefix will be checked as withdrawn, but not removed from the set of monitored prefixes. This way, future announcements for this prefix will still be processed by BHAS.

Check hijack
This routine is entered whenever an the announcing AS of an update matches a monitored prefix its origin AS, but no further suspicious activity was observed. Therefore, the update is considered to be legitimate. It might be that an attacker has withdrawn a hijack. To verify this, the set of hijacks is checked if there exists a non-withdrawn hijack h related to p. If this statement is evaluated to be true, these hijacks are marked as withdrawn. If not, the update is discarded.

Chapter 5

Experimentation

The first section will gain insight in to the experimental environment used for this research, a hypothesis per test is given for which the results are briefly examined in section 5.3. The next part discusses the real-world environment for BHAS. The results chapter shows the actual coverage of IRR records for Dutch Autonomous Systems and their prefixes. The last part sums up the outcome of all tests performed and the results of the real BGP feed.

5.1 Experimentation environment

5.1.1 Topology

The BGP Hijack Alerting System will be tested in a virtual environment which is completely isolated from the Internet. A real-world representative scenario is simulated using a real-world AS topology and corresponding prefixes. This way, various hijacking scenarios can be simulated. Since a real-world topology is used, this environment also fully complies with the RIPEstat API, which would not be the case when creating a testbed using a random topology with RFC1918 networks.

As to be seen in figure 5.1, the testing environment corresponds to a real world peering topology. However, router A101 was added to the topology which contains ExaBGP feeding its received announcements to BHAS.

5.1.2 Technical details

All of the ten routers in the simulation environment use the Quagga router engine. Although ExaBGP has been selected as the preferred router to trigger BHAS execution, Quagga is more router-like and is comparable to Cisco routers in terms of configuration syntax (table 4.3).

These routers were distributed over two physical machines installed with Ubuntu 15.04, both simulating five LXC instances on which Quagga was installed. As to be seen in figure 5.1, the A[*] routers were simulated on machine A, while the B[*] routers were simulated on machine B.

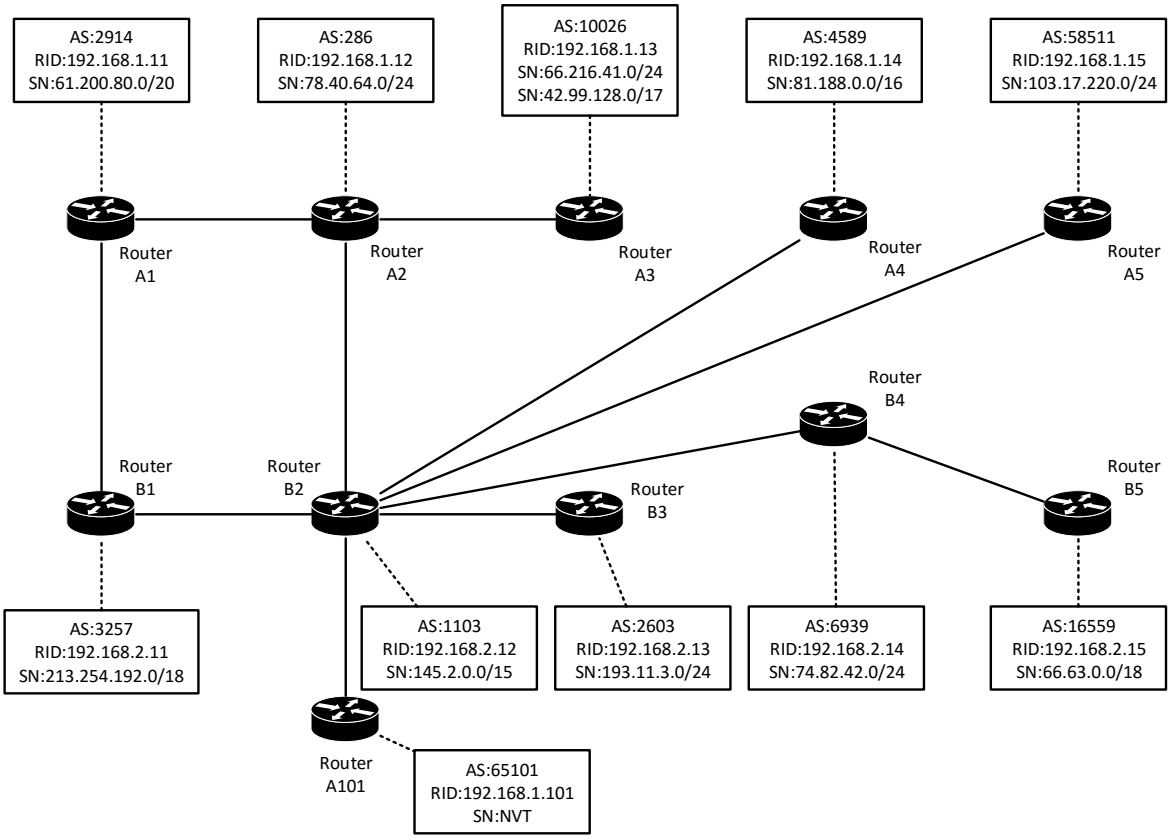

Figure 5.1: BHAS test environment, artificial testbed

5.1.3 Test procedure & hypothesis

All test scenarios described in this section will be conducted in the test environment shown in 5.1. All of the subnets displayed in this figure are being monitored by BHAS. In chapter 4.1.1, five types of hijacks are presented. All of these hijacks were tested. An hypothesis is also described for every test scenario.

Type 1 - prefix hijack
For a type one hijack a router needs to announce a prefix owned by a different AS. Router with AS 3257 will announce the prefix of AS 16559 which is 66.63.0.0/18. In this case the path to the hijacked network is shorter so the announcement will be forwarded to Router A101. BHAS needs to create a prefix hijack notification (type 1).

After the hijack notification the test proceeds with cancelling the hijack. AS 3257 will stop announcing the hijacked network and a withdrawal of this prefix will be sent to AS 1103. The router with AS 1103 shall look into its BGP table to check if has

an alternative route to 66.63.0.0/18. This route is available following AS-path (6939, 16559). This new route will be inserted into the routers routing table and will also be announced to its neighbors. BHAS will receive this new announcement and will consider the hijack to be cancelled as it received a legitimate route. The database should contain one hijack record with the corresponding announce and withdrawal timestamps.

Type 2 - subnet hijack
In this scenario a router needs to announce subnet of a prefix owned by another AS. AS 16559 will announce network 145.2.0.0/16 which is a subnet of a prefix owned by AS 1103. This is known as a more specific route so it will be advertised over the complete topology. When BHAS receives the announcement it needs to register a prefix hijack for the monitored prefix. When AS 16559 stops announcing the hijacked network a withdrawal is sent for this subnet. The saved alert should now be updated with an added withdrawal timestamp.

Type 3 - AS & prefix hijack
Autonomous System 4589 changes its ASN to 16559 and therby starts announcing prefix 66.63.0.0/18. AS 1103 will notice the shorter path to AS 16559 and will prefer the malicious route. Upon receiving the new route, BHAS will notice the new AS path differentiates from the legitimate one. The upstream AS also changed. As the country code of the new upstream should not match the geolocation of the origin AS, BHAS should create a type 3 hijack alert for prefix 66.63.0.0/18. When the hijack is cancelled, BHAS receives the original route and shall set the withdrawal timestamp for the related hijack for this prefix.

Type 4 - AS & prefix hijack
Like the type 3 hijack, AS 4589 is still mimicking AS 16559. In this scenario it solely propagates a more specific prefix, namely 66.63.59.0/24. As this route propagates among all peers, BHAS needs to raise a type 4 hijack for prefix 66.63.0.0/18. Upon withdrawing the more specific prefix, the monitoring system sets the withdrawal date for the hijack.

Type 5 - less specific hijack
AS 10026 owns and advertises 66.216.41.0/24, while AS 16559 owns and advertises 66.63.0.0/18. In this scenario, AS 16559 withdraws its entire prefix and starts announcing subnets of this prefix. So they now announce 66.63.59.0/28 and 66.63.59.128/25, leaving some IP space assigned to this AS unannouned. AS 10026 now starts advertising 66.0.0.0/8. All traffic destined for a destination in 66.0.0.0/8 will go to AS 10026 except traffic destined for the subnets advertised by AS 16559 since the more specific will be preferred for those destinations. The subnets owned by 16559 from 66.63.0.0 until 66.63.63.255 are now available for AS 10026 and could be used for malicious activities. When this less specific reaches the monitoring system it needs to save a type 5 hijack alert into the database for the monitored prefix, which is 66.63.0.0/18. As soon as AS

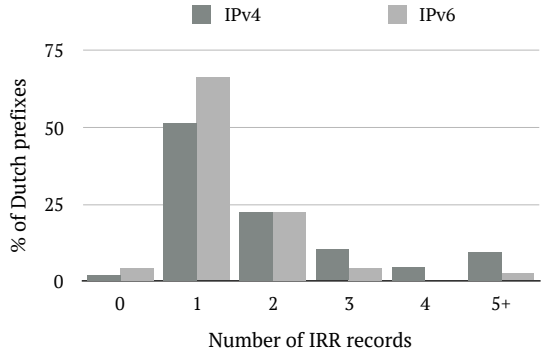

Figure 5.2: Dutch AS IRR coverage

No. of origins	IPv4	IPv6
0	87	31
1	2408	475
2	1054	161
3	476	30
4	212	2
5+	428	17

Table 5.1: Dutch AS IRR Coverage

10026 withdraws the 66.0.0.0/8, BHAS needs to update the hijack entry in the database and set the withdrawal timestamp.

5.2 Real-world environment

BHAS will be connected to a full BGP feed from a Dutch IPS called SURFnet. Thereby it monitors all prefixes registered by Dutch ASes. Information collected for the IRR registration test already contains such a list. In total this concerns 5379 prefixes, both IPv4 and IPv6. We expect BHAS to fill the prefix table from the input during the initialization process. When the initialization is done, the BGP peer should be brought up and the origins table will be enriched with data, like AS county code, origin upstream AS and AS path. When the full feed is received, updated must be processed by BHAS to detect hijacks. In the event of a hijack BHAS needs to register a BGP hijack in the hijack table using the correct type like the model in chapter 4 describes.

5.3 Results

Results regarding the current Dutch IRR coverage will be discussed first, whereafter the experimentations of the simulated hijack and the real-world detection results shall be examined.

5.3.1 IRR coverage

As discussed in chapter 4.1.2, the issued ASNs of all Dutch ASes were collected, whereafter all currently announced prefixes by those ASes were queried from RIPEstat. This resulted in a total of 4664 IPv4 and 715 IPv6 prefixes. In total, 98% of all Dutch IPv4 prefixes are covered by at least one IRR record, whereas 96% of currently announced Dutch IPv6 prefixes have at least one IRR record containing an authorized origin AS (figure 5.2). As to be seen in table 5.1, this comes down to a total of 118 networks in a total set of 5379 prefixes.

5.3.2 Experimentation environment simulations

The test results are displayed in figure 5.2. Every simulated hijack was detected by BHAS. As soon as a hijack was cancelled and its withdrawal reached the monitoring system, the *withdrawn_at* attribute for that hijack was set. For hijack type three and four it should be noted that BHAS detected the invalid change regarding the upstream provider, which was AS 1103 instead of AS 6939.

prefix_id	**subnet**	**mask**	**origin_as**	**origin_upstream_as**	**as_path**	**hijack_type**	**withdrawn_at**
66.63.0.0/18	66.63.0.0	18	3257	1103	1103,3257	1	2016-01-22 13:41:09
145.2.0.0/15	145.2.0.0	16	16559	6939	1103,6939,16559	2	2016-01-22 13:58:14
66.63.0.0/18	66.63.0.0	18	16659	1103	1103,16659	3	2016-01-22 13:43:35
66.63.0.0/18	66.63.59.0	24	16659	1103	1103,16659	4	2016-01-22 13:48:11
66.63.0.0/18	66.0.0.0	8	10026	286	1103,286,10026	5	2016-01-22 14:48:40

Table 5.2: Experimentation environment simulations, reported hijacks

Type 1 - prefix hijack
The router of AS 1103 picked up the update of AS3267 annoucing AS16559 his prefix, and noticed the shorter AS-path to the prefix. The new route was inserted into the routers routing table and a new announcement was send to all peers except the origin. BHAS is a peer for AS1103 and receives the update. The prefix is matched and the originating AS is compared. As expected BHAS recognized the different announcing AS. AS3257 is not allowed to announce this subnet. So the MOAS check also fails. Therefore, an entry was made in the database for a prefix hijack. When AS3257 stopped announcing the hijacked prefix is sent a withdrawal which was received by the router in AS1103. The router checks the BGP table for a backup route to the prefix. It will find one (the original announcer) and inserts that route into the routing table. AS1103 sends an update to all it peers except on the interface it received the announcement. BHAS received the update, the origin AS matches the one in the database and the hijack entry in the database is marked as withdrawn.

Type 2 - subnet hijack
Autonomous System 16559 announces a more specific of a prefix owned by AS 1103. This announcement was forwarded by all routers because they do not have a route to this specific subnet. Since a router will always prefer a more specific route over a less specific, the is propagated until all peers have received it. BHAS matches this subnet to a prefix of interest. The announcing AS is not in the database as an origin AS. The RIPEstat IRR check also fails because AS 16559 is not registered as an authorized origin AS for 145.2.0.0/15. BHAS adds this event as a type 2 hijack to the database. When AS 16559 withdraws the hijack, it's propagated to all peers as no router has an alternative route to this subnet. BHAS received the withdrawal and the hijack entry in the database is marked as ended.

Type 3 - AS & prefix hijack
When former AS 4589 start mimicking AS 16559 and also announces its prefixes, AS 1103 receives a shorter path to 66.63.0.0/18 and updates it routing table. It sends an update for the prefix to all his peers. BHAS will notice the originating AS equals an authorized AS. It compares both paths and notices a change as the upstream AS is different. A geolocation comparison is performed between the new upstream and the announcing AS. Since none of AS 4589 his country codes matches an origin CC of 66.63.0.0/18, the upstream is considered invalid thus a type 3 hijack is created. When only advertising a subnet it will be registered as a type 4. When the hijack stops all prefixes are withdrawn and AS 1103 will act the same way it did as in the prefix hijack part. An update is send to BHAS and the hijack entry is updated.

Type 4 - AS & subnet hijack
The observed behavior for this hijack is similiar to a type 3 hijack. The only exception is that BHAS observed a type 4 hijack instead of a type 3 since it evaluated the announcement to be a more specific prefix of a monitored network.

Type 5 - supernet hijack
As soon as AS 10026 advertises 66.0.0.0/8, AS 1103 will forward it in the same way as a more specific. BHAS marks it as a supernet of a monitored prefix. Since the announcing AS does not equal an authorized origin of the monitored prefix the IRR records are queried. This still doesn't result in an IRR record allowing AS 10026 to authorize this prefix so the event is registered as a type 5 hijack. As soon as BHAS notices the supernet is withdrawn is succesfully sets the withdrawn timestamp indicating the hijack has run its course.

5.3.3 Real-world tests

During the five days that BHAS was connected to a full BGP feed it processed a total of 10.5 million prefix either getting announced or being withdrawn. BHAS monitored all Dutch prefixes as discussed in section 5.4.1. As for the first hour BHAS received the

	IPv4	IPv6
Number of ASes	52814	10581
Total prefixes	589566	26456
Largest AS	AS4538	AS3651
Prefixes largest AS	5594	454
Updates per hour	44824	23676
Announcements	43070	17103
Withdrawals	1754	6573

Table 5.3: Prefix information

full BGP table and processed all updates to fill the AS path column in the Origins table for all 5379 prefixes. After the first hour the average load drops from 682500 updates to an average of 85000 updates per hour. 62% of these updates concerned IPv4 routing information while the remaining 38% were IPv6 updates. Figure 5.3 shows the amount of prefixes getting either announced or withdrawn during the first day that BHAS was online.

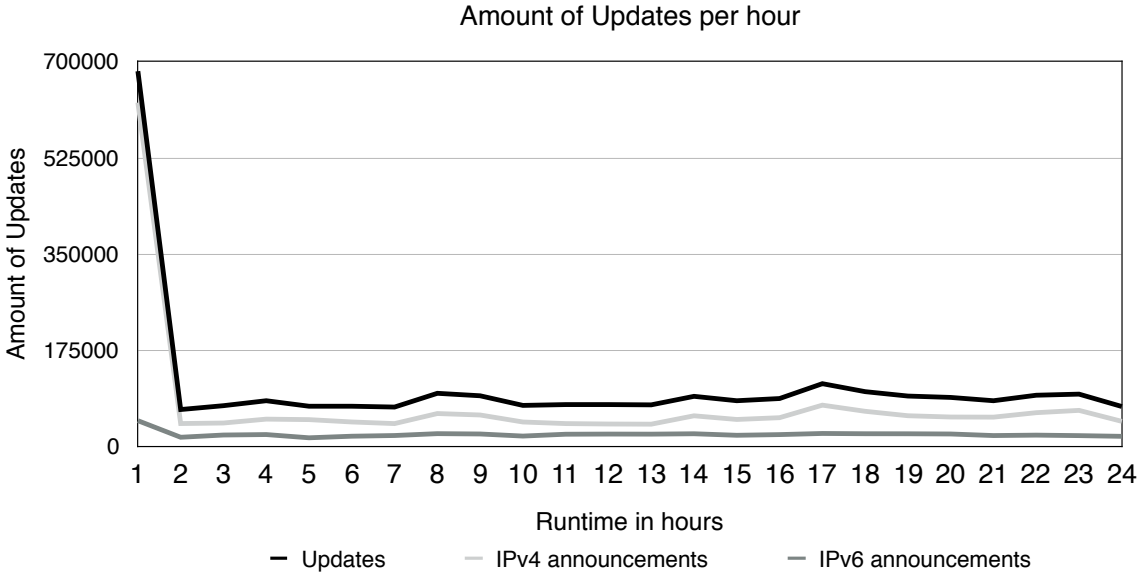

Figure 5.3: Amount of updates per hour processed by BHAS on a full BGP feed

Figure 5.4 shows the total amount of network announcements that were considered interesting, i.e. the announced prefixes were related to monitored prefixes. During the first hour the peak is so high it flattens the entire graph, therefore it is left out. A total of 6494 Dutch prefix announcements and 49 Dutch prefix withdrawals were processed during the first 24 hours. Figure 5.5 shows the amount of interesting withdrawals for the first 23 hour.

As displayed in figure 5.6, during a period of five days 1460 hijacks have been reported

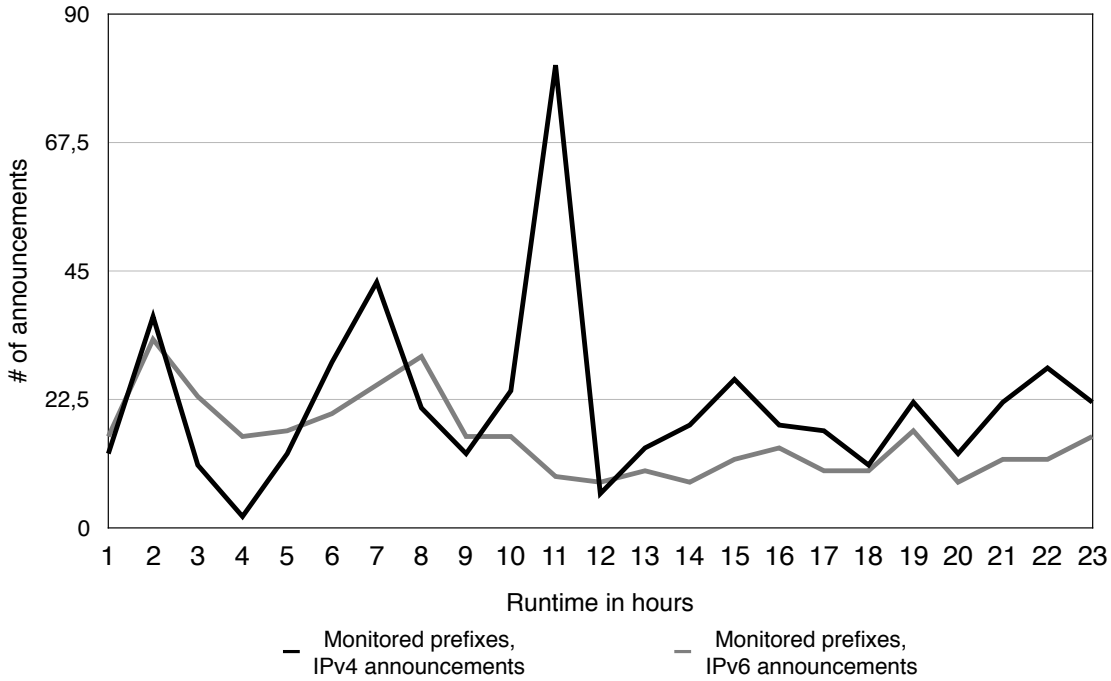

Figure 5.4: Amount of announcements for Dutch prefixes per hour processed by BHAS on a full BGP feed

by BHAS with an average hijack rate of 9 per hour. Within this period 62% of all hijacks have been marked as withdrawn. Prefix hijacks of type 1 and 3 are the most dominant type of hijack which make up 75% of all hijacks. A significant difference is notable between these dominant hijack types. While 93% of all AS & prefix hijacks (type 3) have been witdrawn, only 22% of the prefix hijacks (type 1) have done so. Furthermore, the supernet hijack was not frequently observed and only covers 3% of all reported hijacks.

5.4 Discussion

5.4.1 IRR coverage

Where previous research has shown the IRR coverage of Dutch prefixes was approximately 70%, it can be concluded this has significantly improved to an overall coverage of 96% for both IPv4 and IPv6. However, this does not tell us anything regarding the accuracy, which will be discussed later in this section. Notable is the almost 10% of prefixes using at least five IRR records which are often very fragmented. An extreme outlier is ING (AS15625) who assigned 816 IRR records to different subnets within their 16 bit prefix. A reason for doing so might be that such Autonomous Systems acquired a lot of relatively small prefixes over the past years. This growth is a likely explanation

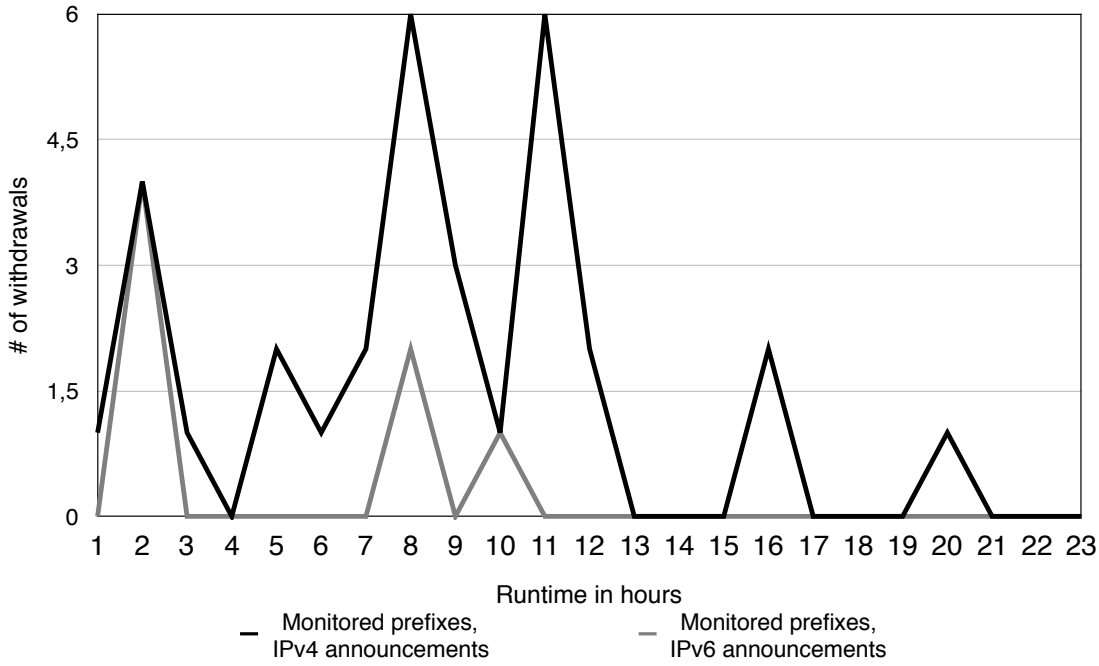

Figure 5.5: Amount of withdrawals for Dutch prefixes per hour processed by BHAS on a full BGP feed

for ING as they started off with a 145.221.24/22 prefix back in 2000[27], for which the 145.221.0.0/16 they possess now is a supernet.

The proper IRR coverage of Dutch ASes does not guarantee an accurate registration of their prefixes and origin ASes. The 52% of prefixes with only one origin AS registered in IRR might only be connected to a single ISP, but they could just as well not have updated their registration. Judging from this data, 47% of all currently announced Dutch prefixes are connected to the Internet in a redundant fashion.

5.4.2 Experimentation environment simulations

BHAS is capable of detecting all types of hijacks described by *Hu et al* as well as the supernet defined by this paper in chapter 4. When subnets are sold and advertised by a different AS BHAS is capable of detecting this by checking authorized origins in the IRR-records.

Although BHAS passed all test scenarios it is not clear from these tests how it would handle real-world behavior. First, hijacks that don't reach the upstream provider which BHAS is connected to won't be detected. This is caused by the nature of BGP which instructs routers to only advertise preferred routes to their neighbors. However, the detection rate of BHAS will certainly improve when connected to multiple BGP feeds, especially when they reside in different continents.

Also note that performing an AS, type 3/4 hijack in the real-world is not viable in

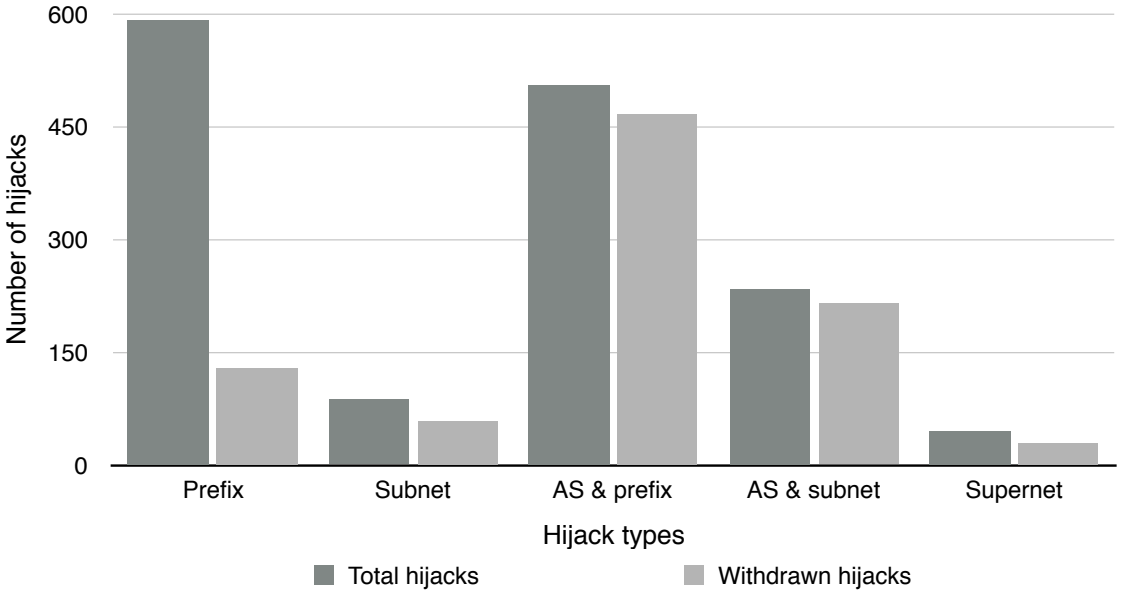

Figure 5.6: Amount of hijacks reported by BHAS after running 124 hours

controlled environments. Since BGP peering needs to be configured on both sides of the peering session. Therefore, the neighbor or upstream provider must agree on the configured AS numbers. On the other hand this could be argued since the majority of all BGP hijacks are non-malicious, but caused by configuration errors instead [18].

5.4.3 Real-world tests

The large number of prefix hijacks could be caused by lacking IRR accuracy. Since the majority of the type 1 hijacks have a permanent nature it's likely they are false-positives due to lacking IRR accuracy, since another AS is announcing a prefix for which they're not registered to do so. For example, BHAS registered a hijack for the University of Amsterdam their 145.109.0.0/17 because it is announced by AS1124 while, according to the IRR registration, it should be announced by AS1103. This point us to the conclusion that BHAS only works in a proper way when the prefixes of interest all have the correct IRR registration. Most of the type 3 hijacks are probably caused by failovers to ISP's residing in a different country than the origin AS. Since failovers are usually short-term, this explains the high number of hijack withdrawals for AS and prefix hijacks.

Performance characteristics of BHAS indicate that the architecture can scale to monitoring a large number of prefixes. Upon receiving the initial BGP feed when the peering with SURFnet is established, the CPU has an average load of 30% (Intel(R) Xeon(R) CPU E3-1220L V2 @ 2.30GHz) while the memory consumption peaks at point around one Gigabyte. Utilizing the CIDR datatype of PostgreSQL is crucial, since the proof of concept could only process around three announcements per second when filtering monitored prefixes manually for every incoming announcement.

Chapter 6

Conclusion

Since the current BGP implementation allows anyone with access to a BGP feed to announce arbitrary prefixes BGP hijacks will happen on a daily basis. Therefore the need for a proper BGP monitoring system exists. Such a system should be able to alert on all five types of hijacks. Online services are available for owners of a small number of prefixes, but using these services cause data leakage. Other solutions are only able to detect hijacks for active subnets and suffer from a high false-positive rate. One solution tries to detect hijacks from the clients sides perspective the majority of the solutions look into data-plane or control-plane data. Most control-plane applications offer near real-time detection and cause minimal overhead. Data-plane solutions can greatly improve the detection accuracy, but have a significant overhead and alerts are often generated with a delay.

Some prefixes have multiple IRR registrations and could be advertised by multiple ASes. These MOAS prefixes could raise a lot of false positives when the system doesn't recognize them in a proper way. Large organizations often use multiple different ISP's to connect to the Internet for redundancy purposes which is also known as multi-homing. This means that the AS-path changes on an ISP failover and could trigger an AS hijack alert for most systems. This is obviously not a real hijack but a false positive.

Organizations needing to monitor a number of prefixes, have concerns about leaking valuable information to external services, are in need of MOAS and multi-homing support and don't want to rely on commercial vendors. There was no available solution meeting these requirements until now. The BGP Hijack Alert System provides a reliable, stealthy way of detecting the four known types of hijacks and detects the hijack introduced in this paper. Along with the support for IPv4 and IPv6 prefix monitoring make BHAS a proper BGP hijack alert system. A new algorithm utilizes public available information such as geolocations and the IRR registry in order to detect BGP hijacks. Leveraging a BGP feed enables near real-time detection of prefix hijacks.

6.1 Discussion

6.1.1 Strengths

Hijack types
BHAS is capable of detecting five different types of hijacks by comparing the prefix from the update to all prefixes in the database and decides if this is a subnet, a prefix match

or a supernet of a prefix within the database. This way BHAS is able to detect hijacks by a less specific route announcement.

Registered updates
Organizations are allowed to sell IP space. Before a hijack type 1, 2 or 5 is alerted, BHAS will always check the RIPEstat API to see if the change of announcing AS is a valid one. When the change is valid the database with monitored prefixes will be updated with the new records.

In 2015 a foreign ISP (UPC) bought a Dutch ISP (ZIGGO). They operate under the same name and could start advertising all the owned prefixes under the same ASN. This will result in an anomaly in the monitoring system. Some networks will be announced by a different AS and by a different router, many systems will detect this as a prefix hijack because these systems work with a baseline and don't query external sources for valid changes. For customers of the ISP owning a ASN the AS-path will be changed to the 'new' ASN as upstream AS. This could result in a AS hijack alert when monitoring tools support this feature. BHAS will check the RIPEstat database when a prefix and ASN doesn't match in the database to see if there is a registration of the change. So the example of the 2 Dutch ISP's, the changes need to be registered in the RIPEstat database. When the administration is correct BHAS will not alert for a prefix Hijack and will update the monitored prefix table. When the upstream AS is changed in the AS-path BHAS checks for the geolocation of the new upstream and compares this to the geolocation of the announcing AS. When these match the system considers this as a valid change like a multi-homed network should work.

MOAS support
Multiple Origin Autonomous System (MOAS) prefixes will not trigger an alarm in the BHAS model. The monitoring database will be filled with all possible announcing origin ASes. Some networks are allowed to be announced by 30 ASes according to RIPEstats IRR information.

Multi-home support
Large companies often choose to connect their network to the Internet by multiple ISPs. This is called Multi-home. BHAS is capable of detecting this anomaly in the AS-path by comparing the country code of the new upstream AS to the country code of the announcing AS.

IPv4 and IPv6 support
BHAS detects IPv4 and IPv6 hijacks on all five types of hijacks. The algorithm does not make a difference between IPv4 or IPv6 prefixes.

6.1.2 Limitations

This model is limited by the amount of BGP feeds it receives. In the case of an AS hijack or a full prefix hijack on the other side of the world the system will never raise an alarm when it's connected to only one BGP feed. A BGP neighbor will not re-announce a route to a prefix when it has a more preferred route to it. It will save the update in its BGP table but will never forward it to a neighbor. When a route to a network is learned with a longer path than the original it will not forward it to neighbors. To make this system more effective it should have BGP peers with ISP's all over the world.

The model that is used as the fundamentals to create BHAS supports the use of Multi Originating AS (MOAS) but they must have an IRR registration. Some articles are written about the state of the IRR registration[32,31] stating that about 46% of the information is accurate. It turns out that 96% of the Dutch prefixes is registered. There is no way of telling if these registrations are correct.

6.2 Future work

Validate hijack alerts BHAS generates hijack alerts since it is connected to the live BGP feed. It is unknown if these alerts are all valid. The amount of generated false positives and false negatives should be studied and determined how to lower these, if necessary.

Compare to other solutions It would be a good test to see how well BHAS performs in comparison to other hijack monitoring tools. Looking into detection speed and the number of false positives and false negatives would be very interesting.

Other sources of information BHAS is now depending on a full BGP feed. But there are more sources of information to utilize and to improve the results of the algorithm. Some of the sources have a delay[35] where other sources provide near real-time feeds in a different format[42]. Previous studies showed a large number of looking glasses accessible over the Internet of which the routing table can be downloaded without disclosing prefix information. In what way could these different sources be added as a data source for BHAS?

	BHAS	Description
Detect prefix hijack [2]	Yes	With one BGP feed BHAS is limited in detecting full prefixes.
Detect prefix and AS hijack [2]	Yes	With only one BGP feed BHAS is limited in detecting AS and prefix hijacks.
Detect more specific hijack [2,3,16]	Yes	More specifics will always be detected when connected to a full BGP feed
Detect more specific and original AS hijack [2,16]	Yes	More specifics will always be detected when connected to a full BGP feed
Detect hijack of unused prefixes [18]	Yes	BHAS is able to detect hijacks of unused prefixes
Detect valid change of AS [15,16]	Yes	By doing a RIPEstat query on a possible hijack BHAS will check for IRR registration updates
Support for Multi Origin AS (MOAS) [3]	Yes	Monitored prefixes will be stored in the database with all registered ASes
Detection delay [16]	Max 40 seconds	It takes 40 seconds before a BGP update is distributed across the world
Overhead [16]	full BGP feed, RIPEstat queries	BHAS needs a full BGP feed to work and RIPEstat queries are send in the case of possible hijacks
Stealty [2]	Yes	BHAS can be implemented by using a snapshot of the RIPEstat database. This will introduce a delay
Disclose prefixes to the public	Yes	RIPEstat API is used in case of possible hijacks.
Identify attacker [16,15]	Yes	BHAS will report the announcing AS and Upstream AS of a hijack.

Table 6.1: BHAS is compared to the same rich feature set as the other solutions referred in chapter 2

Bibliography

[1] Y Rekhter, T Li, and S Hares. Rfc 4271: Border gateway protocol 4, 2006.

[2] Xin Hu and Z Morley Mao. Accurate real-time identification of ip prefix hijacking. In *Security and Privacy, 2007. SP'07. IEEE Symposium on*, pages 3–17. IEEE, 2007.

[3] Changxi Zheng, Lusheng Ji, Dan Pei, Jia Wang, and Paul Francis. A light-weight distributed scheme for detecting ip prefix hijacks in real-time. In *ACM SIGCOMM Computer Communication Review*, volume 37, pages 277–288. ACM, 2007.

[4] Bert Koenders. Beantwoording kamervragen over kapen ip-adressen. `https://www.rijksoverheid.nl/regering/inhoud/bewindspersonen/bert-koenders/documenten/kamerstukken/2015/08/26/beantwoording-kamervragen-over-kapen-ip-adressen`, 2015.

[5] Chinese isp hijacks the internet. `http://www.bgpmon.net/chinese-isp-hijacked-10-of-the-internet/`. Accessed: 2016-01-25.

[6] Youtube hijacking: A ripe ncc ris case study. `https://www.ripe.net/publications/news/industry-developments/youtube-hijacking-a-ripe-ncc-ris-case-study`. Accessed: 2016-01-25.

[7] Hitesh Ballani, Paul Francis, and Xinyang Zhang. A study of prefix hijacking and interception in the internet. In *ACM SIGCOMM Computer Communication Review*, volume 37, pages 265–276. ACM, 2007.

[8] bgpmon.net (bgp monitoring service). `http://www.bgpmon.net/`. Accessed: 2016-01-25.

[9] Dyn.com/renesys (bgp monitoring service). `http://dyn.com`. Accessed: 2016-01-25.

[10] Mohit Lad, Daniel Massey, Dan Pei, Yiguo Wu, Beichuan Zhang, and Lixia Zhang. Phas: A prefix hijack alert system. In *Usenix Security*, 2006.

[11] Jong Han Park, Dan Jen, Mohit Lad, Shane Amante, Danny McPherson, and Lixia Zhang. Investigating occurrence of duplicate updates in bgp announcements. In *Passive and Active Measurement*, pages 11–20. Springer, 2010.

[12] Ioannis C Avramopoulos and Jennifer Rexford. Stealth probing: Efficient data-plane security for ip routing. In *USENIX Annual Technical Conference, General Track*, pages 267–272, 2006.

[13] Ernst Biersack, Quentin Jacquemart, Fabian Fischer, Johannes Fuchs, Olivier Thonnard, Georgios Theodoridis, Dimitrios Tzovaras, and Pierre-Antoine Vervier. Visual analytics for bgp monitoring and prefix hijacking identification. *Network, IEEE*, 26(6):33–39, 2012.

[14] He Yan, Ricardo Oliveira, Kevin Burnett, Dave Matthews, Lixia Zhang, and Dan Massey. Bgpmon: A real-time, scalable, extensible monitoring system. In *Conference For Homeland Security, 2009. CATCH'09. Cybersecurity Applications & Technology*, pages 212–223. IEEE, 2009.

[15] Zheng Zhang, Ying Zhang, Y Charlie Hu, Z Morley Mao, and Randy Bush. Ispy: detecting ip prefix hijacking on my own. In *ACM SIGCOMM Computer Communication Review*, volume 38, pages 327–338. ACM, 2008.

[16] Xingang Shi, Yang Xiang, Zhiliang Wang, Xia Yin, and Jianping Wu. Detecting prefix hijackings in the internet with argus. In *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, IMC '12, pages 15–28, New York, NY, USA, 2012. ACM.

[17] Bgpmon github, saif el-sherei. `https://github.com/ssherei/BGPmon`. Accesed: 2016-01-27.

[18] Pierre-Antoine Vervier, Olivier Thonnard, and Marc Dacier. Mind your blocks: On the stealthiness of malicious bgp hijacks. 2015.

[19] Xiaoliang Zhao, Dan Pei, Lan Wang, Dan Massey, Allison Mankin, S Felix Wu, and Lixia Zhang. An analysis of bgp multiple origin as (moas) conflicts. In *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, pages 31–35. ACM, 2001.

[20] The shape of a bgp update. `https://labs.ripe.net/Members/vastur/the-shape-of-a-bgp-update`. Accessed: 2016-01-14.

[21] Cidr report, ipv4 prefix report. `http://www.cidr-report.org/as2.0/`. Accessed: 2016-01-26.

[22] Cidr report, ipv6 prefix report. `http://www.cidr-report.org/v6/as2.0/`. Accessed: 2016-01-26.

[23] Geoff Huston and Grenville Armitage. Projecting future ipv4 router requirements from trends in dynamic bgp behaviour. In *Proc. of ATNAC*, 2006.

[24] H Berkowitz, E Davies, S Hares, P Krishnaswamy, and M Lepp. Terminology for benchmarking bgp device convergence in the control plane. Technical report, 2005.

[25] Ipv4 prefix number plot. `http://www.cidr-report.org/cgi-bin/plota?file=%2fvar%2fdata%2fbgp%2fas2.0%2fbgp-active.txt&descr=Active%20BGP%20entries%20%28FIB%29&ylabel=Active%20BGP%20entries%20%28FIB%29&with=step`. Accessed: 2016-01-26.

[26] Ipv6 prefix number plot. http://www.cidr-report.org/cgi-bin/plota?file=%2fvar%2fdata%2fbgp%2fv6%2fas2.0%2fbgp-active.txt&descr=Active%20BGP%20entries%20%28FIB%29&ylabel=Active%20BGP%20entries%20%28FIB%29&with=step. Accessed: 2016-01-26.

[27] Ripe stats. https://stat.ripe.net/.

[28] Maxmind geoip. http://dev.maxmind.com/. Accessed: 2016-01-07.

[29] Giuseppe Di Battista, Tiziana Refice, and Massimo Rimondini. How to extract bgp peering information from the internet routing registry. In *Proceedings of the 2006 SIGCOMM Workshop on Mining Network Data*, MineNet '06, pages 317–322, New York, NY, USA, 2006. ACM.

[30] Download ripe database. https://www.ripe.net/manage-ips-and-asns/db/faq/faq-db/can-i-download-the-ripe-database. Accessed: 2016-01-26.

[31] Route hygiene: The dirt on the internet. http://research.dyn.com/2009/03/compliance-scoring-by-country/. Accessed: 2016-01-25.

[32] How accurate are the internet router registries (irr). http://www.bgpmon.net/how-accurate-are-the-internet-route-registries-irr/. Accessed: 2016-01-25.

[33] Geoff Huston, T Bates, and P Smith. Cidr report. *Web site, http://www. cidr-report. org*, 2005.

[34] Beichuan Zhang, Raymond Liu, Daniel Massey, and Lixia Zhang. Collecting the internet as-level topology. *ACM SIGCOMM Computer Communication Review*, 35(1):53–61, 2005.

[35] The university of oregon route views. http://www.routeviews.org/. Accessed: 2016-01-15.

[36] IETF RFC 6396, October 2015.

[37] Ying Zhang, Zheng Zhang, Zhuoqing Morley Mao, Charlie Hu, and Bruce MacDowell Maggs. On the impact of route monitor selection. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 215–220. ACM, 2007.

[38] Quagga. http://www.nongnu.org/quagga/. Accessed: 2016-01-04.

[39] Exabgp software router. https://github.com/Exa-Networks/exabgp. Accessed: 2016-01-04.

[40] Bird software router. http://bird.network.cz/. Accessed: 2016-01-04.

[41] Postgresql cidr datatype. http://www.postgresql.org/docs/9.1/static/datatype-net-types.html. Accessed: 2016-01-31.

[42] Bgpmon.io. http://www.bgpmon.io. Accessed: 2016-02-03.

[43] Open bgpd. http://www.openbgpd.org/. Accessed: 2016-01-04.

[44] Xorp software router. http://www.xorp.org/. Accessed: 2016-01-04.

[45] Ipinfusion - zebos. http://www.ipinfusion.com/about/press/ip-infusion-helps-network-equipment-manufacturers-transition-software-defined-networking. Accessed: 2016-01-04.

[46] Extensive list of bgp tools, o'reilly. http://www.bgp4.as/tools. Accessed: 2016-01-04.

[47] bgptools. http://nms.lcs.mit.edu/software/bgp/bgptools/. Accessed: 2016-01-04.

[48] Modeling of bgp. http://inl.info.ucl.ac.be/system/files/emanics-summer-school-talk.pdf. Accessed: 2016-01-04.

[49] Internet routing registry. http://www.irr.net/. Accessed: 2016-01-19.

[50] Ripe asn allocation. ftp://ftp.ripe.net/ripe/stats/delegated-ripencc-latest. Accessed: 2016-01-25.

[51] Iana asn allocation. http://www.iana.org/assignments/as-numbers/as-numbers.xml. Accessed: 2016-01-25.

[52] Cidr report asn allocation. http://www.cidr-report.org/as2.0/autnums.html. Accessed: 2016-01-25.

[53] Sandra Murphy. Bgp security vulnerabilities analysis. 2006.

[54] Bgp instability report. http://bgpupdates.potaroo.net/instability/bgpupd.html. Accessed: 2016-01-26.

[55] Christian Horn. Understanding ip prefix hijacking and its detection. In *Seminar internet routing, intelligent networks (INET)*, 2009.

Appendix A

	BGPmon[8]	Dyn.com[9]	Saif El-Sherei bgpmon.py[17]	Hu et al.[2]	PHAS[10]	iSPY[15]	Zheng et al.[3]
Detect prefix hijack[2]	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Detect prefix and AS hijack[2]	Yes	Yes	Yes	Yes	Yes, Routeviews database	Yes	No
Detect more specific hijack[2,3,16]	Yes	Yes	Yes	Yes	Yes	No	Yes
Detect more specific and original AS hijack[2,16]	Yes	Yes	Yes	Yes	Yes	No	No
Detect hijack of unused prefixes[18]	Yes	Yes	Yes	No	No	No	No
Detect valid change of AS[15,16]	No	No	No	No	No	Yes	No
Support for Multi Origin AS (MOAS)[3]	Yes	No	No	Yes	No	Yes	Yes
Detection delay[16]	Max 40 seconds	Max 40 seconds	Max 40 seconds	40+ seconds	3 hours delay	Max 40 seconds	Depends on traceroute database and amount of networks monitored
Overhead[16]	Closed source	Closed source	Full BGP feed, ASN tool	Full BGP feed, ICMP traffic to monitored networks and TCP handshake	Routeviews DB, ICMP traffic to monitored networks	ICMP and TCP traffic to Internet	ICMP traffic to monitored networks
Stealty[2]	Closed source	Closed source	Yes	No	No	No	No
Disclose prefixes to the public	Yes	Yes	Yes, ASN tool query	No	No	No	No
Identify attacker[16,15]	Yes	Yes	Yes, except for AS hijacks	Yes	Yes	No	No

Table A.1: Comparison of all useful products that could be used for BGP monitoring, products are compared on a rich feature set. Some of them are closed source and do not share information about these specifics

Appendix B

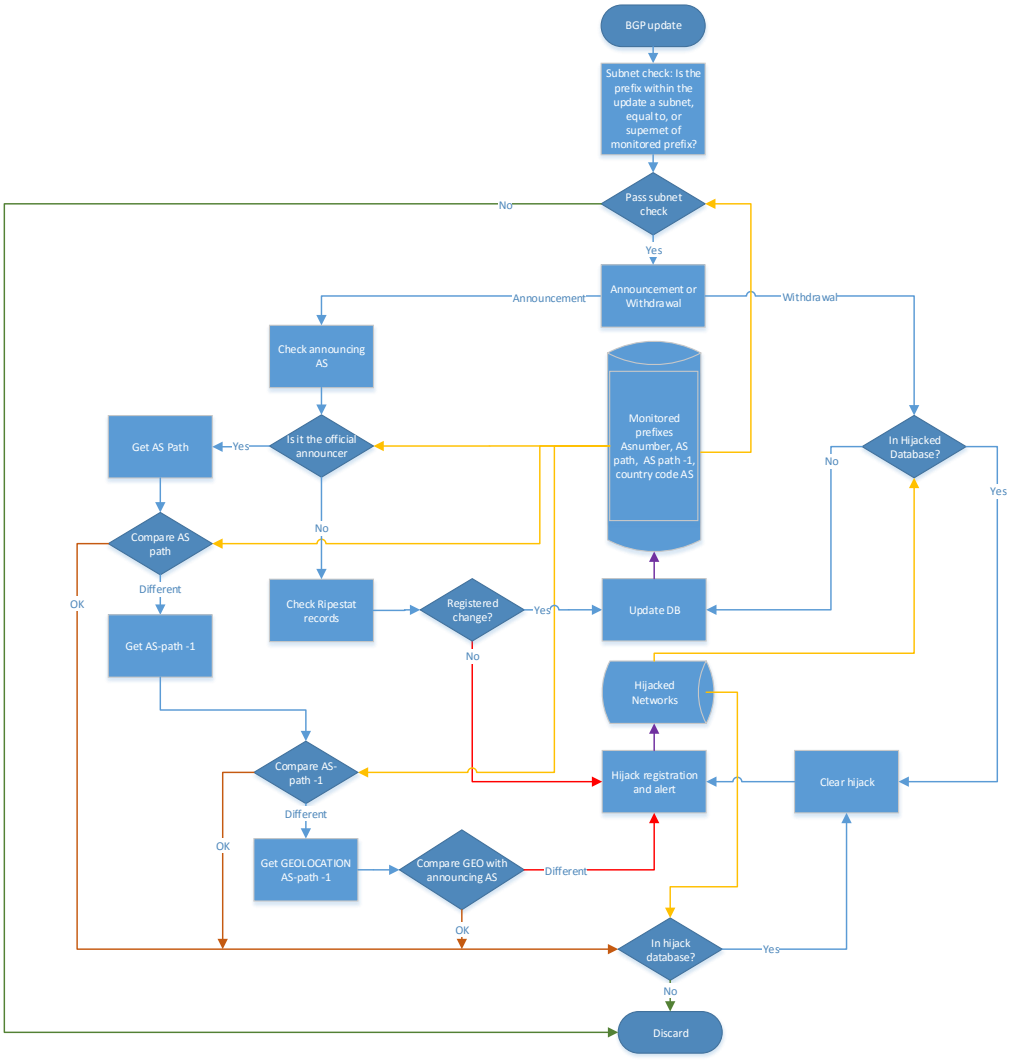

Figure B.1: Hijack Detection Algorithm, flowchart