ABSTRACT

XIAOLIANG ZHAO. Toward a Fault-Tolerant Border Gateway Protocol. (Under the direction of Professor Shyhtsun Felix Wu.)

Today, the Internet has become the nerve center of our society. However, the Internet has been faulty, insecure, unreliable and unavailable, which causes much financial loss and many security problems. Studies show that the current de facto inter-domain routing protocol, Border Gateway Protocol (BGP), is vulnerable to various attacks, and routing-based attacks have unfortunately become quite feasible. Enhancing the fault-tolerance property of BGP is a very important and timely issue for the sake of overall Internet robustness and security. At least in the short term, it is very hard to deploy a new protocol or upgraded version of BGP within today's Internet because BGP has been widely used for years. Therefore, this dissertation focuses on providing practical solutions to existing problems by using existing BGP mechanisms and fault detection techniques. Challenges exist because BGP only propagates aggregated information instead of raw information; the dynamics of BGP are difficult to understand; there is no common operational practice, and the coordination and cooperation between different administrative domains is hard to achieve.

In this dissertation, BGP vulnerabilities have been analyzed from different perspectives. A set of critical BGP-related problems has been identified. One of them is called the Multiple Origin Autonomous System (MOAS) problem. A practical enhancement to BGP is presented to enable BGP to distinguish valid MOAS cases due to operational needs and invalid MOAS cases caused by faults. One key property of this solution is believed to be its resilience against any single point of failure. Solutions are also provided to solve other problems under the same framework and operations provided by BGP . Equally important, solutions have been fully evaluated against real BGP data or via simulations. The evaluation results show our solutions are very effective.

Toward a Fault-Tolerant Border Gateway Protocol

by

Xiaoliang Zhao

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Computer Sciences

Raleigh

2002

APPROVED BY:

Wenke Lee, Ph.D.	Douglas S. Reeves, Ph.D.
Carla D. Savage, Ph.D.	
Arne A.Nilsson, Ph.D. Co-Chair of Advisory Committee	Shyhtsun Felix Wu, Ph.D. Co-Chair of Advisory Committee

To

Dad, Mom, and Kai

PERSONAL BIOGRAPHY

Xiaoliang Zhao was born in Shandan, Gansu Province, P. R. China. He attended Nankai University in Tianjin from 1991 to 1995 where he graduated with a B.S. degree in Computer Science. He entered the graduate school of the Chinese Academy of Sciences immediately thereafter where he obtained his M.S. degree in Computer Science. Xiaoliang came to the USA for his Ph.D. study in the Department of Computer Science at North Carolina State University in Fall, 1998. Since then, he has worked as a research assistant in Secure Highly-Available Network Group (SHANG), under the supervision of Dr. S. Felix Wu. His research interests include Intrusion Detection Systems (IDS), network management, and fault-tolerant routing protocols. Since August 2000, he has been actively involved in a DARPA-funded project, Fault-Tolerant Networking Through Intrusion Identification and Secure Compartments (FNIISC). With others, he identified the Multiple Origin Autonomous System (MOAS) problem and proposed a practical solution. He also worked on other solutions to enhance the fault tolerance property of BGP and overall Internet security. He is an author and co-author of several conference/workshop papers and Internet drafts.

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank my advisor, Dr. Shyhtsun Felix Wu, who introduced me into an exciting research area, guided me through the 4-year study and taught me how to become a good researcher. I have always believed there are only a few magic events which will change one's life. When Dr. Wu opened the door for me to come to the USA for my Ph.D study and brought me into the FNIISC project, I think such an event really happened to me.

I sincerely thank my committee members, Dr. Wenke Lee, Dr. Arne A. Nilsson, Dr. Douglas S. Reeves and Dr. Carla D. Savage, for their great support and constructive comments. Dr. Reeves not only helped me on my research, but also helped me to take care of many administrative issues.

I want to give special thanks to my colleague, Dr. Daniel Massey, for his consistent support and many one-to-one discussions during my visit at USC/ISI East. With the invaluable help from him, I made my first breakthrough for my research. Besides, he and his wife generously provided me with accommodations during my stay in Washington, D.C.

I really feel fortunate to work with an excellent team, which includes top-class researchers like Prof. Lixia Zhang (UCLA), Allison Mankin (USC/ISI), Randy Bush (AT&T), Dr. Felix Wu and others. Their broad knowledge and quick insights to hard problems have been impressive and enlightening. Their attention to details has been challenging but productive. It will be more proper to view the work described in this dissertation as a result of this collaboration

effort instead of one individual's contribution.

I also would like to thank all of my colleagues, Ho-Yen Change, Zhi Fu, He Huang, Dan Pei, Feiyi Wang, Lan Wang, Chien-Lung Wu, Jim Yuill and countless others. They have helped me in so many ways that it is impossible for me to enumerate them. I would say that I've had my greatest time to work with them and to be a friend of them.

Last, also the most important, I would like to express my deepest gratitude to my parents, although they may not even understand these English words I was writing here. They have committed themselves to bringing up eight kids during a very hard time and giving their full support to our college educations. I owe my largest debts to them. I also want to thank my wife, Kai, for her encouragement, understanding and endurance. I would have never been able to go through this process without her love and support. I would like to dedicate this dissertation to them.

TABLE OF CONTENTS

LIST OF FIGURES **x**

LIST OF TABLES **xiii**

1 Introduction **1**

1.1 Internet Security Overview 1

1.2 Border Gateway Protocol (BGP) Tutorial 2

1.2.1 Operational Environment 3

1.2.2 Basic Operation 4

1.2.3 BGP Message Format 5

1.2.4 Handling UPDATE Message 8

1.2.5 Finite State Machine of BGP Process 9

1.2.6 Routing Information Reduction 10

1.2.7 Current Practices 11

1.3 Research Objectives and Challenges 15

1.4 Dissertation Contributions 20

1.5 Reader's Guide 22

2 BGP Vulnerability Analysis **24**

2.1 Vulnerabilities Inherited from Protocol Design 24

2.2 Potential Threats and Costs 30

2.2.1 Threat from Outsiders 30

2.2.2 Threats from Insiders 31

2.2.3 Costs 32

2.3 Related Work 33

2.4 Case Studies 37

2.4.1 Multiple Origin AS (MOAS) 38

2.4.2 Origin AS Change (OASC) 40

2.4.3 Slow Convergence 44

2.4.4 Malicious/Unstable Routing Path 45

3 MOAS/OASC **47**

3.1 MOAS Measurement and Analysis 47

3.1.1 Measurement Method 48

3.1.2 Measurement Results 49

3.1.3 Classification of MOAS cases 52

3.1.4 Explanations and Implications of MOAS 54

3.1.5 Re-examine MOAS Durations 58

3.2 Detecting Invalid MOAS 60

3.2.1 MOAS List 60

3.2.2 Implementing the MOAS List in BGP 62
3.2.3 Limitations of the Proposed Solution 65
3.2.4 Identifying the Correct Origin AS 66
3.2.5 Simulation 68
3.3 Detecting OASC Anomalies 77
3.3.1 OASC Measurement 78
3.3.2 A Visual-Based Approach 80
3.3.3 Anomalies Detected 87
3.4 Chapter Summary 89

4 Slow Convergence **95**

4.1 Assertions for Improving BGP Convergence 95
4.1.1 Basic Assertions 96
4.1.2 Practical Considerations 99
4.2 Simulation Results 103
4.2.1 Simulation Setup 103
4.2.2 Route Failure 104
4.2.3 Route Failover 107
4.2.4 Discussion 110
4.3 Chapter Summary 112

5 Malicious/Unstable Routing Path **113**

5.1 Motivations and Path Filtering Approach 113
5.2 Design of Path Filters 117

5.2.1 A Simple Path Filter . . . 119
5.2.2 An Adaptive Path Filter . . . 121
5.3 Evaluation . . . 122
5.3.1 Parameter Setting . . . 122
5.3.2 Data Source and Methodology . . . 124
5.3.3 Impacts on Server Reachability . . . 129
5.3.4 Effectiveness . . . 134
5.4 Chapter Summary . . . 136

6 Conclusions and Future Work **138**

6.1 Dissertation Contribution Summary . . . 139
6.2 Future Work . . . 141

Bibliography **143**

LIST OF FIGURES

1.1 Interconnection Example 3

1.2 Network Reachability and Topology Information Propagation . . 5

1.3 Finite State Machine of BGP Process 10

1.4 Multi-homing Example 12

1.5 Traffic Engineering Example 14

1.6 BGP message storm of September 18 - 19[COPY01]. 16

2.1 Topology without faulty routers 28

2.2 Topology with 2 faulty routers 28

2.3 S-BGP at Aggregation Point 35

2.4 Originating a BGP Route 39

2.5 Prefix With Two Valid Origin ASes 40

2.6 An Incorrect Origin AS 41

2.7 Route Spoof Attack Example 46

3.1 The number of MOAS cases from 11/1997 to 07/200 50

3.2 Duration of MOAS cases 51

3.3 Distribution among prefix length 52
3.4 Distribution of MOAS Classes 54
3.5 Example scenario 63
3.6 Example MOAS List 63
3.7 Example configuration file 63
3.8 Simulation Topology - 63 AS 69
3.9 Spoof-Resilience of Our Scheme in the 46-AS Topology 72
3.10 Comparison between 25-AS, 46-AS and 63-AS Topology 74
3.11 Comparison between Partial and Complete Deployment of MOAS Detection 76
3.12 OASC Data Process 78
3.13 OASC Data 79
3.14 Quadtree coding of IP prefixes 82
3.15 Visualization of data 84
3.16 Data on a typical day 86
3.17 Data on August 14, 2000 92
3.18 CSM-type changes on April 18, 2001. 93
3.19 CMS-type changes on April 19, 2001 94

4.1 Example network topology 98
4.2 Topology statistics for network with 30 nodes 104
4.3 Comparison of original and enhanced BGP for route failures . . . 106
4.4 Comparison of original and enhanced BGP for route failovers . . 109

4.5 Comparison of cumulative convergence percent for route failure and route failover 111

5.1 Route stability to top level DNS servers 115

5.2 Route stability to top level DNS servers 116

5.3 Simulated Routers 127

5.4 Calculation of Servers Reachability 129

5.5 Adaptive Path Filter's Impacts on Servers Reachability 130

5.6 Simple Path Filter's Impacts on Servers Reachability 133

LIST OF TABLES

1.1 BGP Message Header 6

1.2 UPDATE Message format 6

2.1 A Route Object Example 36

2.2 Slow Convergence in the Internet 45

3.1 Median of MOAS cases per year 49

3.2 Expectation of the duration of MOAS cases 51

5.1 Path Changes along the Time 115

5.2 Parameter Setting 124

5.3 RRC0's peering ASes that we examined 125

5.4 Percentage of Time when N or More DNS Root Servers are Reachable 131

5.5 Slow Convergence (AS3549) 135

Chapter 1

Introduction

1.1 Internet Security Overview

After its birth in 1969 and its commercialization in late the 1980's, the Internet has rapidly grown into a giant network, which today connects more than one hundred million computers [Zak]. The Internet is everywhere, from home to offices, from companies to governments and from hospitals to universities. It has greatly influenced our daily life. However, the Internet has been faulty, insecure, unreliable and unavailable, and this has caused financial loss and security problems. The following are just a few examples:

- In 1988, the first Internet worm was spread on the Internet. It affected over 6,000 out of 60,000 hosts [Boe00] and resulted in severe Internet connectivity problems.

- On April 25, 1997, a routing software bug caused a major portion of the Internet to melt down for at least 20 minutes. It was estimated that up to 40 percent of Internet users were affected [B+97].

- On Feb. 2000, Yahoo.com was out of service for three hours due to a heavy Distributed Denial of Service (DDoS) attack. Some major sites such as Amazon, CNN, eBay, etc were the victims as well [Jeo01].

- On July 2002, the Code Red virus infected around 359,000 hosts in less than 14 hours [Moo01], and also caused a lot of network congestion.

The objective of this dissertation is to study the issues related to improving overall Internet security, especially from the global routing infrastructure perspective. In general, the global routing system security has been overlooked in the past few years, but it is an undoubtedly important subject.

1.2 Border Gateway Protocol (BGP) Tutorial

The Internet, as its name suggests, is a network of networks. Each network connects with one or more other networks at the shared points, such as Public Exchange Points (EP) [ep], or via dedicated links. The Inter-Domain Routing (IDR) protocols are used to exchange network reachability and topology information between networks, so that eventually every router knows how to forward data packets to the correct destination. Border Gateway Protocol version 4 (BGP) [RL95] is the current widely deployed Inter-Domain Routing protocol. Intuitively, BGP can be viewed as the protocol which glues different networks

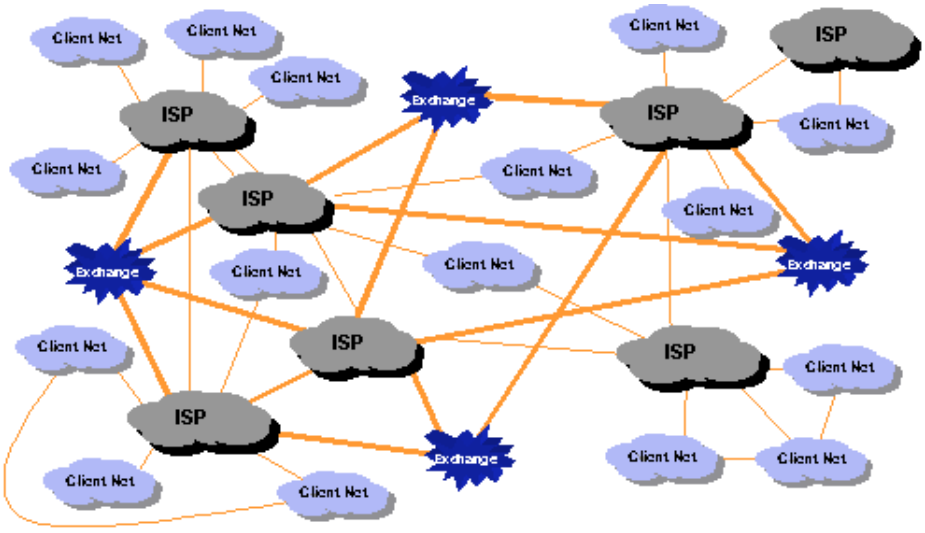


Figure 1.1: Interconnection Example [Hus00], where ISP stands for the Internet Service Provider, or provider for short, who provides the Internet connectivity for the client networks.

into one giant, the Internet. An example interconnection of different networks is shown in Figure 1.1.

1.2.1 Operational Environment

Each network within the Internet is identified by its IP address prefix. For example, the North Carolina State University (NC State) campus network is identified as 152.1.0.0/16[1], which means every host in the NC State campus network shares the same first 16 bits. An Autonomous System, or *AS* for short, is loosely defined as networks and routers with the same routing policy control. Each AS is assigned a unique AS number. For example, the AS number for the

[1]NC State may use other prefixes as well, but for simplicity, we only use 152.1.0.0/16 as an example

North Carolina Research and Education Network (NCREN)[2] is 81. Thus the network 152.1.0.0/16 is under the administrative control of AS 81. Informally, we could also say AS 81 owns the IP prefix 152.1.0.0/16.

Routers executing BGP protocol are commonly referred to as *BGP routers* or *BGP speakers*. Two connected BGP routers can establish a BGP session and exchange the network reachability and topological information via *BGP messages* or *BGP updates*. One router in a BGP session is commonly referred to as a *BGP peer* or *BGP neighbor* of its correpondent. A BGP session is differentiated into two forms: *eBGP session* and *iBGP session* based on whether the two BGP routers belong to different ASes or to the same AS. Two ASes that have established a eBGP session are called neighboring ASes.

1.2.2 Basic Operation

The basic operation of BGP involves an AS announcing its own prefixes to all its neighboring ASes. According to the local policy, a neighbor can decide whether or not to accept these announcements and propagate them to its own neighbors. For example, in a network as shown in Figure 1.2, AS 1 owns the network, represented by its prefix, p. AS 1 announces its reachability to p to its two neighbors, AS 2 and AS 3. After receiving and accepting such announcement, AS 2 and AS 3 know p can be reached via AS 1. In turn, AS 3 announces its reachability to p to its own neighbors, AS 2 and AS 4, but not AS 1. So does AS 2. This propagation process will continue so that eventually every AS will know

[2]NC State campus network is a part of NCREN

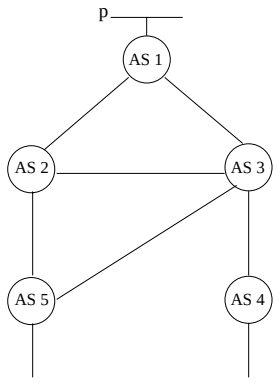


Figure 1.2: Network Reachability and Topology Information Propagation

how to reach p[3]. It also might be the case that AS 2 chooses not to propagate AS 1's information according to its local policy. Thus AS 5 only can learn p from AS 3.

1.2.3 BGP Message Format

There are four different types of BGP messages, namely, OPEN, UPDATE, NOTIFICATION and KEEPALIVE. In short, the OPEN message is used to setup a BGP connection; the UPDATE message is for exchanging the network reachability and topological information; the NOTIFICATION message is used for notifying the peer of the occurrence of errors, and the KEEPALIVE message acts like a heart beat signal to keep a BGP session alive. All of the four message types share the same header, as shown in Table 1.1, where the numbers in the first row indicate the length of the field in bytes.

[3]Or p's aggregated form if aggregation occurs

16	2	1	variable
Marker	Length	Type	Data

Table 1.1: BGP Message Header

The *Marker field* is designed to contain the value which can be used to authenticate a peer; however, many implementations leave this field blank. The *Length field* gives the total length of the BGP message in bytes. The *Type field* indicates one of the four types we just mentioned above and is followed by BGP data.

For simplicity, we only describe the format of UPDATE message, shown in Table 1.2.

2	variable	2	variable	variable
Unfeasible Routes Length	Withdrawn Routes	Total Path Attribute Length	Path Attributes	Network Layer Reachability Information

Table 1.2: UPDATE Message format

The UPDATE message is the most important message because most routing information is transported by this type of message. Roughly, an UPDATE message contains two different types of content. The first two fields contain the information of those networks which are no longer reachable. The last three fields contain the information of those networks which can be reachable, as indicated by the field of *Network Layer Reachability Information*, and their associated attributes, termed as *Path Attributes*.

So far, there are 8 different types of *Path Attributes*. The most important one is the *AS_PATH*, which includes a list of ASes through which a network can be reached.[4] For example, the BGP route "152.1.0.0/16: (5056 1 209 81)" means that from AS 5056's point of view, the IP prefix 152.1.0.0/16 could be reached by first going to AS 1, then to AS 209, and finally to AS 81. The last AS in an AS path is referred as the *Origin AS* for that prefix. In the above example, AS 81 is the Origin AS for the IP prefix 152.1.0.0/16. An AS in the middle of an AS path is referred as the *Transit AS* because it provides the data transit service for its two neighboring AS in the same AS path. In the above example, AS 1 provides the transit service for AS 5056 and AS 209. The AS path is crucial in BGP because it represents how to reach a particular prefix and prevents potential loops and also is used to enforce routing policies.

The other commonly used path attributes include:

- *ORIGIN* defines where the routing information was learned from. It could be one of the three values: IGP (from Interior Routing Protocol, such as OSPF), BGP (from a BGP peer), and INCOMPLETE (from configuration).

- *NEXT_HOP* defines the IP address of the next router on the path to the destination.

- *MULTI_EXIT_DISC* is used by an AS to indicate the preferred router when the incoming traffic can enter the AS from multiple border routers.

[4] In the case of route aggregation, an element in the AS path may include a set of ASes.

- *LOCAL_PREF:* is used by an AS to indicate the preferred path for the outgoing traffic when there are multiple paths available for the same destination.

- *COMMUNITY_ATTRIBUTE:* provides a mechanism for a BGP router to pass additional information to both peering neighbors and remote BGP routers in order to facilitate and simplify the control of routing information.

However, the current BGP specification does not mandate any security protection for AS path, origin AS, and other attributes. Any router in the middle of the propagation path could manipulate the information arbitrarily without being detected. The consequence of such manipulation may be very serious though.

1.2.4 Handling UPDATE Message

When a BGP router receives a new UPDATE message, first the router will apply any local policies to decide whether or not it should accept the routing information inside the message. For example, as a security enhancement, many service providers allow their customers to only announce their allocated networks; any other networks announced from the customers will not be accepted.

If allowed, the prefixes which are reachable and their associated AS paths will be stored in a particular database, *Adj_RIB_in*.[5] For each reachable prefix,

[5]If there already exists an entry for the same prefix, the new route will replace older one.

Adj_RIB_In contains a set of feasible paths learned from different peers. Similarly, the corresponding entry for unreachable prefixes will be removed from *Adj_RIB_In*.

Then the BGP router will run its decision process to select the best path from *Adj_RIB_In* and re-advertise its best path to other peers, if the local policy allows the path to be advertised. The selected best path will be stored in another database, called *Adj_RIB_Local*. It is possible that the newly selected best path is the same as the previous selection, and in this case, the best path will not be re-advertised.

The best path selection process takes the changed entry in *Adj_RIB_In* as the input and examines the associated path attributes in a specific order. *LOCAL_PREF* is the first attribute to be evaluated; the largest *LOCAL_PREF* will be favored. If all routes have same value for *LOCAL_PREF*, then the route with shortest length of *AS_PATH* will be preferred. We can continue this tie-breaking procedure by orderly examining the value of *ORIGIN* and *MULTI_EXIT_DISC* and so on. The final metric used to break ties is the router ID, which is a global unique value.

1.2.5 Finite State Machine of BGP Process

Each BGP process runs as a finite state machine, shown in the Figure 1.3. Basically, the BGP process will move from the initial Idle state to the Connect state when an operator starts the BGP session via console. Depending on the underlying transport status and the messages it receives, the BGP process will

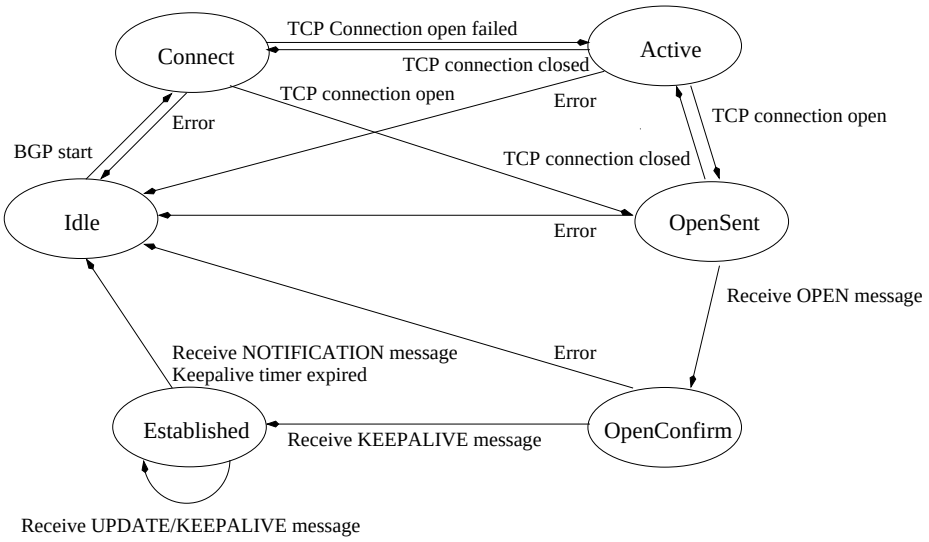


Figure 1.3: Finite State Machine of BGP Process

change its state until the Established state is reached. At this state, the BGP process will start to exchange routing information by sending and receiving BGP UPDATE messages.

1.2.6 Routing Information Reduction

To reduce the routing information, a set of prefixes could be aggregated into one prefix under certain circumstances. For example, 152.1.2.0/24 and 152.1.3.0/24 could be aggregated to 152.1.2.0/23. In this example, we say 152.1.2.0/24 is more specific than 152.1.2.0/23 and conversely, 152.1.2.0/23 is less specific than 152.1.2.0/24. Aggregation is intended to reduce the routing information as well as to provide some degree of localization of routing changes; however, with the increased adoption of multi-homing, as described in the following section, more

and more more-specific prefixes are exposed to the global routing system.

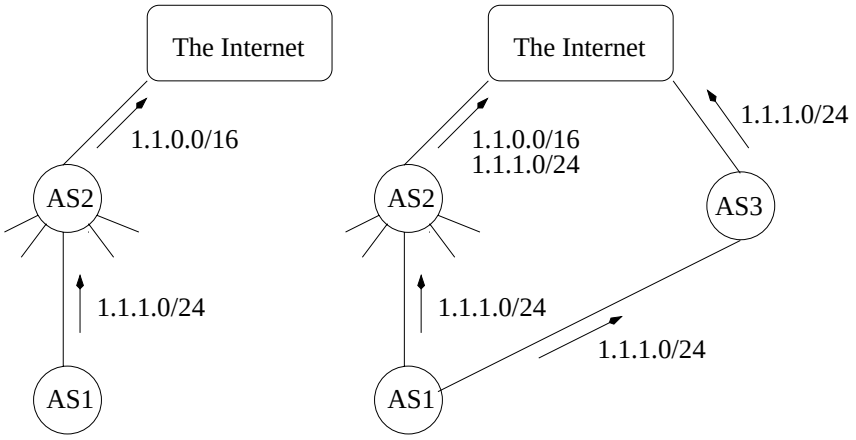
1.2.7 Current Practices

Multi-homing

With the recent decrease of connectivity costs, it has become affordable for many networks, especially small networks, to purchase Internet connectivity from more than one provider. By multi-homing, those networks not only increase available bandwidth, but also improve their resilience to providers' failures. In other words, when one provider fails to provide Internet connectivity, the data traffic still can be delivered through other providers' networks.

The benefits from multi-homing are so attractive that it has quickly become a common practice. However, the payoff is that the aggregation mechanism becomes less effective to reduce the network routing information and localize faults. To illustrate this point, we show an example.

In Figure 1.4, AS 2 provides the Internet connectivity to AS 1 and allocates the subnet 1.1.1.0/24 to AS 1. But in Figure 1.4(a), AS 2 is the only provider of AS 1, while in Figure 1.4(b), AS 1 has two providers, AS 2 and AS 3. When AS 1 is single-homed, as shown in Figure 1.4(a), AS 2 only needs to announce 1.1.0.0/16, and traffic destined to both AS 1 and AS 2 will be delivered to AS 2 first. When AS 1 is multi-homed, as shown in Figure 1.4(b), AS 2 may have to announce both 1.1.0.0/16 and 1.1.1.0/24, because otherwise AS 1 will only be reached via AS 3 (AS 3 will announce 1.1.1.0/24 anyway, and BGP prefers more-specific prefixes).



(a) Single-homing scenario (b) Multi-homing scenario

Figure 1.4: Multi-homing Example

The consequences of such multi-homing case are not only one more entry (1.1.1.0/24) will be added into the global routing table[6], but also any instability and fault that occurs in 1.1.1.0/24 will be exposed to the outside world. For example, when the link between 1.1.1.0/24 and the outside world goes up and down, such local events will be propagated to the global Internet.

As pointed out in [Hus01a], recently the Internet routing table appears to be growing exponentially and this might become a great challenge for the Internet routing system: if the exponential growth rate continues, the size of the routing table may eventually exceed the available computational power, thus no router can handle such a large table. Multi-homing also poses security challenges because it may lead to the scenario that one prefix might be originated from

[6]When the global routing table is concerned, it is generally referred to as the BGP routing tables in the Internet core routers

more than one AS under certain configurations, which makes it indistinguishable from the same scenario caused by faults.

Traffic Engineering

Traffic engineering attempts to balance the traffic loads on multiple links. However, due to issues like complex Internet topology, measurement challenges, and data traffic dynamics, traffic engineering practices are not much more than an operational "art", that can be achieved by using Multiprotocol Label Switching (MPLS), tuning IGP and/or BGP parameters, prepending one's own AS number multiple times in a disfavored path, or using other ad-hoc approaches. One commonly used approach is selectively advertising more-specific prefixes to different peers, as illustrated in Figure 1.5.

In Figure 1.5, AS 1 receives data traffic from its two providers, AS 2 and AS 3. However, it is possible that the incoming data traffic is significantly asymmetric, e.g., one link is almost congested by the incoming traffic but the other is free. To balance the traffic load, AS 1 may selectively advertise two more-specific prefixes to two providers; for example, AS 1 may advertise 1.1.1.128/25 to AS 2 and advertise 1.1.1.0/25 to AS 3, as shown in Figure 1.5. Consequently, the incoming traffic destined to 1.1.1.128/25 will be delivered via the link between AS 1 and AS 2, while the rest of the traffic will be delivered via another link.

Such use of BGP will add more smaller prefixes into the global routing table, and growth is further exacerbated.

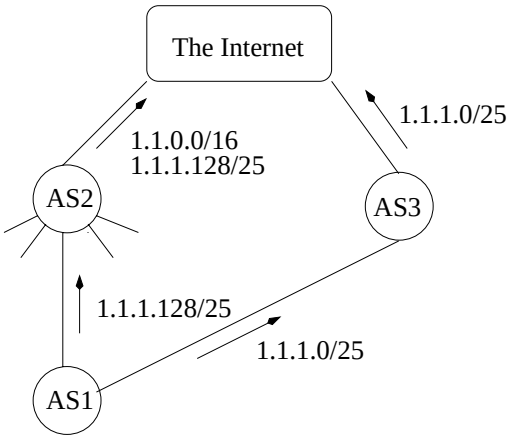


Figure 1.5: Traffic Engineering Example

Policy Routing

BGP sometimes could be viewed as a policy routing protocol because its operation has been greatly influenced by administrative policies. These policies determine what kind of routing information should be accepted or announced, which route should be preferred, or which prefix should be aggregated or deaggregated. Policy increases the overall complexity and dynamics of BGP behavior because BGP could behave totally differently under different policies. Also, policy normally remains as an internal knowledge to an AS and poses great challenges for the analysis of the BGP routing data. Furthermore, Griffin [GW99] pointed out that under certain policies, BGP may not converge at all.

1.3 Research Objectives and Challenges

Today, the Internet has become the nerve center of our society. With more services relying on the Internet, unintentional faults or malicious attacks against network protocols, such as routing protocols, become a serious threat to our Internet-connected society. Recent studies show a trend of attacks against the Internet infrastructures, like DNS services and routing systems [tre02], and such attacks could potentially cause catastrophic failure of Internet services. However, as one of the critical components of the Internet, the routing system is exposed to various attacks without any strong protection [Mur02b].

As the de facto inter-domain routing protocol, BGP has been observed to be pathological and faulty. In 1997, a buggy Bay-Network BGP router falsely deaggregated thousands of network addresses which disabled the whole east coast Internet for up to 12 hours [B$^+$97]. Labovitz et al. [LABJ00] found sometimes BGP may converge very slowly when a route was withdrawn. Griffin [GW99] even showed us that BGP may not converge at all under certain conditions. In the middle of 2001, worm attacks such as CodeRed and Nimda were spread around the Internet. The surprising observation was that these two worm instances affected not only the victim web servers, but also, possibly, the BGP protocol stability due to a report from Renesys [COPY01], as shown in Figure 1.6. Yet, we should feel lucky because, if the attacker successfully hijacked 13 root DNS servers and 13 gTLD servers by announcing false routing information, the whole Internet would have gone.

Kent[KLS00] proposed one approach, commonly known as Secure BGP (SBGP),

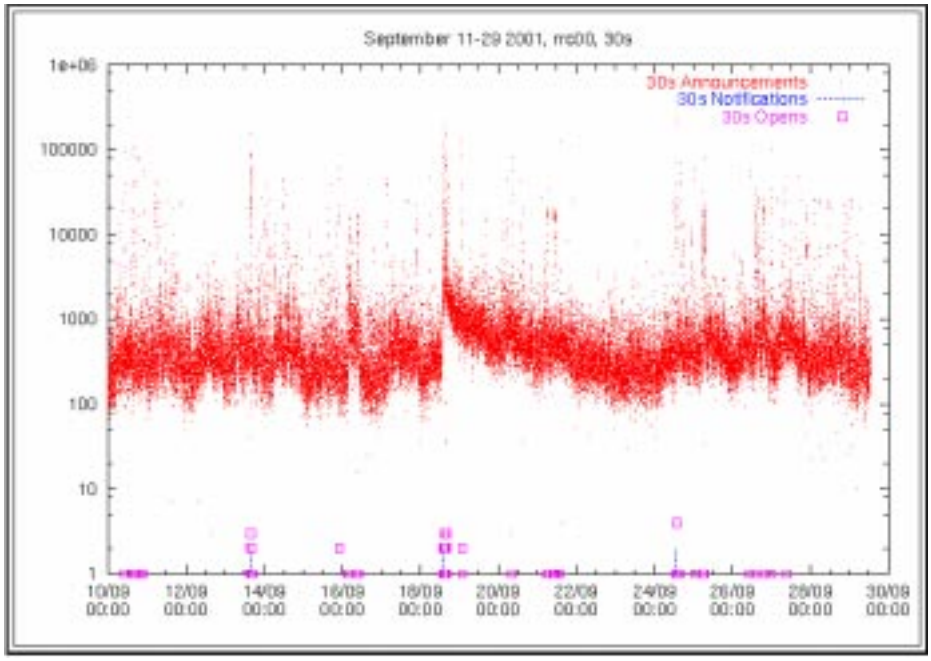


Figure 1.6: BGP message storm of September 18 - 19[COPY01].

to improve the BGP security properties. SBGP uses Public Key Infrastructure (PKI) to authenticate, verify, and authorize route announcements, which does provide some security protection for some routing information elements, such as origin AS and AS path. However, SBGP has the following limitations.

- As Murphy pointed out in [Mur02a], SBGP may encounter difficulties when aggregation occurs.
- SBGP requires significant modifications of protocol and implementations, For example, BGP message format has to be extended to carry additional digital signatures and BGP process has to be modified to verify those signatures.
- SBGP cannot solve other BGP problems like BGP slow convergence and BGP instability due to extreme network conditions.

Until today, there is no proposed inter-domain routing protocol to replace BGP. Even if such a protocol exists, or BGP is modified to be more robust and secure, at least in the short term, it is very hard to deploy a new protocol or upgraded version of BGP within today's Internet because BGP has been widely used for years. Therefore, in this dissertation, we try to provide solutions by using existing BGP mechanisms, such as community attributes, and fault detection technologies. Accordingly, our solutions would be very practical to use. Equally important, we evaluate our solutions against real BGP data to test the effectiveness and impact of our solutions. I believe that provision of practical and effective solutions for the sake of making BGP more resilient to

faults and attacks is the key contribution of this dissertation.

However, there exist some challenges.

- Unlike some routing protocols such as OSPF, which floods the raw link state information throughout the entire network, BGP only propagates the aggregated routing information. In general, with raw information, it would be much easier to detect invalid routing announcements because each router is able to compare the same piece of information received from the different neighbors. Unless all the neighbors were cheating, presumably a rare case, a router should easily detect the inconsistency when a message was altered during the flooding. In this sense, BGP faces more challenges than OSPF does. However, BGP cannot be designed to use a flooding mechanism because flooding makes exorbitant use of network resources which makes itself unable to scale. That is the reason why OSPF normally works in a small network with at most hundreds of routers; comparably BGP is operated within the network consisting of millions of routers.

- In essence, BGP is a simple routing protocol, but when running in a large-scaled network and interacting with other Internet components, such as various other routing protocols, policies and human operators, its behavior turns out to be so dynamic that even understanding BGP behavior could be a challenge. For example, according to the BGP specification [RL95] and its underlying routing algorithm, one could easily think that for a particular destination, an AS should always announce one single best route to

all of its neighbors. However, due to traffic engineering, such assumption might not be always true. Take RFC 1930 [HB96] as another example, according to this document, one may think it should be quite uncommon for a prefix to be originated from multiple ASes. However, our measurement shows that the number of such occurrences is much larger than we expected.

- Although there are some suggested operational guidelines, there are no common BGP operational practices. BGP can be tailored one way or the other to meet various operational needs. Different AS may run different BGP implementations, set different values to BGP parameters, define different policies, and so on. Thus when proposing a new solution, one has to consider if the solution can work within every possible operational setting. For example, BGP community attribute is widely used for various purposes. However, different routers may treat the attribute very differently; some may drop it, but some may keep propagating it. Also as we mentioned earlier, in the normal case, SBGP may work well, but at the aggregation point, it may not work efficiently.

- Other challenges exist because BGP runs in the Internet, which is a large scale, loosely coupled, and continuously growing system, without any centralized control. The only coordination among all the ASes is that they all run the same standard routing protocol. Without a greater degree of coordination and cooperation, improving overall BGP security is very difficult.

1.4 Dissertation Contributions

Recent studies show a trend of attacks against routing infrastructures, and such attacks could potentially cause severe damages. Enhancing the fault-tolerance property of BGP is a very important and timely issue for the sake of overall Internet robustness and security. However, BGP has no strong protection from attacks or unintentional faults. Proposing a new protocol to replace BGP, or deploying enhanced BGP seems not be the best choice in the short term because BGP has been widely used for years. To provide practical and effective approaches to make BGP more resilient to potential attacks and faults, this dissertation works on the solutions which use the existing BGP mechanisms and fault detection technologies. More specifically, in this dissertation, we identified a set of the most critical problems related to BGP and provided solutions to solve these problems under the same framework and operations provided by BGP. Equally important, the proposed solutions are fully evaluated against the real BGP data to test their effectiveness and associated impact. In detail, this dissertation has the following contributions:

1. With the help from others, I identified the Multiple Origin AS (MOAS) as one of the critical BGP-related problem. MOAS could be a problem because on the one hand, MOAS can occur for valid reasons such as multi-homing and on the other hand, router mis-configurations have also produced MOAS. Large scale network outages and other problems have been associated with the MOAS problem. From the standpoint of fault-

tolerance and security, how to determine whether MOAS is the result of a fault or a valid operation is an interesting challenge.

2. To solve the MOAS problem, I proposed to add a bit more information into the existing BGP framework to facilitate MOAS validation. The newly added information for a MOAS is a list of the correct origin ASes of a prefix. The list will be configured beforehand and propagated with the route announcements. If a router receives more than one list for the same prefix, the possible reasons could be configuration errors, either unintentional or intentional, software bugs, or the lists themselves being corrupted. In either case, it would be proper to raise an alarm, and further investigation might be conducted according to the local security policy. This approach also utilizes the rich interconnectivity of the current Internet, which makes the solution resilient against any single point of failure. The reason is because a compromised router can inject false routes into the system, but it cannot easily prevent correct routes from being propagated everywhere; thus other routers can detect the faults by noticing the conflicts between correct and false route announcements.

3. To further validate more complex forms of invalid route announcements from the faulty origin ASes, I started to work on the Origin AS Change (OASC) problem, which attempts to detect the abnormal OASC cases. With the help from information visualization technology, Soon Tee Teoh and I along with others developed a prototype of a visual-based OASC anomaly detection system and identified several anomalies from the his-

torical BGP data.

4. BGP could converge slowly under certain conditions. We proposed to check the existing BGP information to speed up its convergence. This work showed how a simple extra check could improve BGP greatly. To evaluate our idea, I simulated networks with up to 60 ASes, and the simulation results showed that BGP convergence had been greatly improved by adopting our approach.

5. A filtering mechanism is widely used to enhance BGP robustness in practice. We demonstrated how to exploit this simple mechanism to improve the overall Internet security by protecting DNS top-level servers from a route spoofing attack. The route spoofing attack tries to direct legitimate DNS queries to a malicious server by announcing a fake route. Catastrophic failure of Internet could occur when sites are unable to reach DNS top-level servers or sites are convinced to use an impostor server. We proposed to use filters to prevent such attacks. However, filters may adversely impact the reachability to servers when legitimate route change occurs. I analyzed one year of historical BGP data and showed that the impacts are negligible.

1.5 Reader's Guide

The dissertation is organized as follows:

- Chapter 2 analyzes BGP vulnerabilities and introduces several critical

BGP-related problems, namely, MOAS/OASC problem, slow convergence and malicious/unstable routing paths. The current related efforts to enhance BGP security property are also reviewed.

- Chapter 3 introduces the Multiple Origin AS (MOAS) problem and the Origin AS Change (OASC) problem and their solutions;
- Chapter 4 introduces the BGP slow convergence problem and our proposed solution;
- Chapter 5 introduces how to prevent the routing spoof attack against DNS top level servers;
- Chapter 6 concludes the dissertation and proposes some future work.

Chapter 2

BGP Vulnerability Analysis

2.1 Vulnerabilities Inherited from Protocol Design

In this section, we analyze BGP vulnerabilities from the protocol design point of view. There are two major routing protocols: Distance Vector Routing Protocol (DVRP) and Link State Routing Protocol (LSRP). In short, a router executing DVRP computes the shortest distance to every destination and sends out the results, as a vector, to its neighbors (*telling the neighbors what the world looks like*). A router executing LSRP sends the list of neighbors to every other router, normally utilizing a flooding mechanism (*telling the world who the neighbors are*). In essence, BGP is a DVRP, although it is commonly referred to as a Path Vector Routing Protocol, which is considered a variant of DVRP. From

the fault tolerance point of view, these two protocols impose different challenges because DVRP propagates aggregated routing information while LSRP floods raw routing information. Because it is in general harder to detect invalid routing announcements and identify faulty nodes without raw information, DVRP faces more challenges than LSRP.

When we view the BGP system as a large distributed system which consists of a large number of BGP processes running the same protocol, we can conduct the analysis further by applying some results from the field of fault tolerance distributed system. We first introduce the term *Byzantine Failure*, which is an important concept in the field of fault-tolerant distributed systems. This term is derived from a theoretical problem, Byzantine General Problem, first proposed by Lamport et al. [LPS82]. In this problem, a General and n Lieutenants were fully connected and communicated via messengers. The general sent out a command, either "attack" or "retreat", to all lieutenants. All lieutenants further exchanged this command with each other and tried to reach a decision based on the messages they received. However, Several lieutenants or even the general could be traitors in the sense that they could behave in an arbitrary way, such as sending conflicting messages. Now the problem is, in the presence of traitors, finding a solution so that:

1. all loyal lieutenants eventually reach the same decision; and

2. if the general is loyal, all loyal lieutenants should reach the same decision as the general originally sent out.

The Byzantine General Problem turns to be a difficult problem and attracted a lot of research attention. In major research results Lamport [LPS82] showed that the problem is solvable only when fewer than 1/3 are traitors; Fischer [FLP85] proved that there is no solution for an asynchronous system.

In her dissertation [Per88], Perlman classified the faults into two different types: *Simple Failure* means a node or a link failed to operate while a node with *Byzantine Failure* may behave arbitrarily, such as sending conflicting messages, delaying messages, forging messages, or deleting messages. Based on this classification, a level of robustness of a distributed system is defined as:

- *Simple Robustness:* a system operates correctly in the presence of simple failures;
- *Self-Stabilization:* a system can reach a safe state started from an arbitrary initial state;
- *Byzantine Detection:* a system is capable of detecting any nodes with Byzantine failures;
- *Byzantine Robustness:* a system operates correctly in the presence of a limited number of Byzantine failures.

Note that Byzantine Robustness may not imply Byzantine Detection; a system can continue to operate with Byzantine failures but without knowledge about which one is culprit. However, a system with both Self-Stabilization and Byzantine Detection properties is just as good as with Byzantine Robustness,

because after removing the detected faulty nodes, the system still can run to a safe state thereafter.

In [Per88], Perlman designed a Byzantine robust routing protocol based on a flooding mechanism and public key cryptography. This work has been used to design the widely used intra-domain routing protocol: Open Shortest Path First (OSPF) [Moy94]. However, flooding raw routing information, as OSPF does, consumes a significant amount of bandwidth which limits itself and is not suitable in a large-scaled network. In order to be easily scaled, on the contrary, BGP propagates only aggregated routing information. From the fault tolerance point of view, it is much harder to detect invalid routing announcements without raw information. Thus BGP faces more challenges than OSPF.

BGP is desinged to tolerate simple failures, but not Byzantine failures. In his thesis [Mas00], Massey has proved that BGP is theoretically unable to be self-stabilized in the presence of Byzantine failures because BGP doesn't refresh the routes periodically, which has been proven to be a necessary condition for any self-stabilization algorithm.

Furthermore, Massey also pointed out that theoretically it is impossible for the Internet to have a routing protocol with Byzantine Robustness as far as the scalability issue is concerned. The scalability requirement for an inter-domain routing protocol rules out those candidate protocols which either use flooding as the packets delivery mechanism, because flooding makes exorbitant use of network resources, or use a high degree of pre-configured global knowledge, such as a single centralized link state database. Consequently, it is impossible

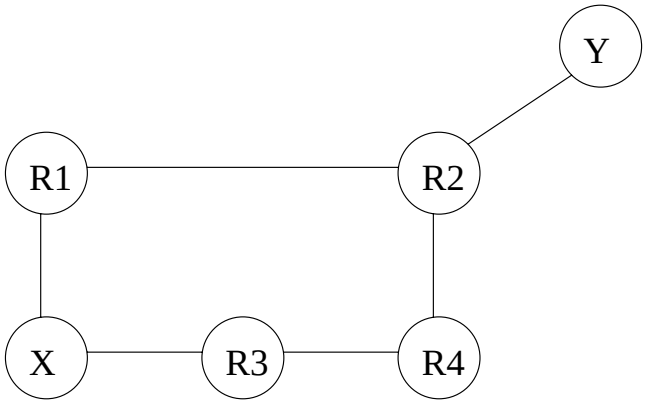


Figure 2.1: Topology without faulty routers

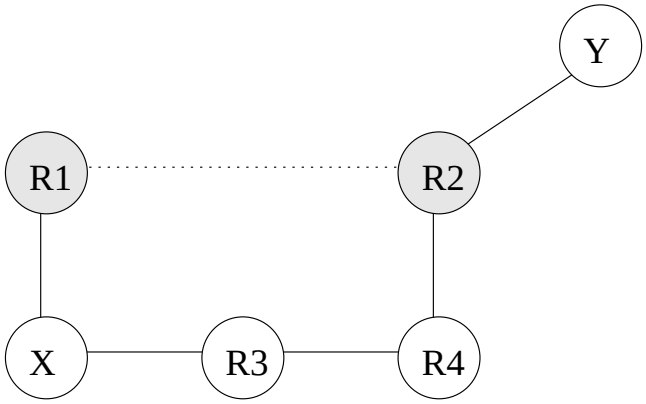


Figure 2.2: Topology with 2 faulty routers

to validate the existence of a link when two ends of this link are both with Byzantine failures. To illustrate this idea, we will show an example.

In the topology without any faulty routers, shown in Figure 2.1, the router X will prefer the path (X R1 R2 Y) to reach Y because it is the shortest path in the graph. In the topology with 2 faulty routers, shown in Figure 2.2, R1 and R2 are faulty and there is no link between them, but both R1 and R2 falsely announce the existence of a link connecting them. In this case, it is impossible

for X to distinguish these two topologies relying on the only information BGP provides[1]. Consequently, X will choose the invalid path (X R1 R2 Y) as the best path. By delivering packets along such path, there is no guarantee all packets can be delivered to the correct destination without being altered, dropped or intentionally delayed. The impossibility for X to distinguish between the faulty topology and correct topology results in the impossibility for BGP to be Byzantine Robust.

Furthermore, one may curious to know why router X could not verify the validity of link between R1 and R2 by just running some sort of non-BGP test, running *traceroute* for example. However, it is also impossible. First, because X doesn't know which link is valid or not, X has to test every link. When considering the size of the Internet and dynamic nature of the Internet topology, it is unfeasible to test the link between every pair of BGP routers in the Internet. Second, delay, traffic load dynamics, temporary hardware failure, etc. all make it extremely difficult to verify the existence of some remote link. R1 and R2 will attempt to lie and make the link appear valid. They may even appear to forward packets across the false link by tunneling themselves over some other path. At worst, this will add some delay but it will look like the link works (for the test traffic). It is impossible to tell if the link between R1 and R2 does not exist or if it is just congested, low-bandwidth, or being attacked by another site to make it appear slow.

[1] Note that with some kind of external knowledge such as a global topology database where every link has to be registered into, X might be able to detect a fake link. However, such centralized database has a scalability problem and may not handle well the dynamics of the Internet topology.

Due to the inherited vulnerability, some BGP routers can announce false routing information easily and remain undetected, which poses the great challenge to BGP fault tolerance study.

2.2 Potential Threats and Costs

Current BGP design and implementation is vulnerable to various network misconfigurations or attacks because BGP has not specified mechanisms to verify the authenticity, integrity, and authority of the announced routing information. The most comprehensive BGP security analysis can be found in [Mur02b] [Mur02a]. In her work, Murphy differentiated the potential threats against BGP into two types: *threats from outsiders* and *threat from insiders*.

2.2.1 Threat from Outsiders

Given it has the capability to access BGP communication channels, an outsider attacker could intercept BGP messages, insert false BGP messages, replay old announcements, etc. Our preliminary experiments have shown the success of such attacks. The lower layer security protection, such as IPSEC [KA98], might provide sufficient protection.

A BGP router operates within a complex and heterogeneous environment, where it interacts with not only BGP routers, but also with routers executing different routing protocols, like OSPF, IS-IS, RIP, etc. Any weak points exist in this environment could be exploited to attack BGP. Our preliminary experiments also show that the OSPF misconfigurations may disrupt BGP routing.

Because the overall routing system security is determined by the weakest point, special considerations have to be taken for the interactions between different protocols.

Another type of outsider attack is the Denial of Service (DOS) attack. Because BGP runs on the top of TCP/IP, any DOS attacks against TCP/IP will also stop BGP from operating. With the recent increase of DOS attacks and the trend of attacking routers [tre02], routers must be protected from such attacks in advance.

2.2.2 Threats from Insiders

A router with Byzantine failure potentially has greater threats than outsiders. The faulty router can forge or manipulate routing information in an incorrect manner and spread the wrong information through legitimate channels. Also it may deviate from the protocol specification, or simply stop functioning, and so on. Software bugs, configuration errors, and malicious attacks could be considered as insider threats.

False route announcements caused by router misconfigurations or software bugs have been observed a number of times over the last few years [Met96] [B+97] [nan98a] [nan98c] [nan01] [cis01]. The most severe Internet outage so far occurred in April, 1997, when a small ISP in the USA falsely deaggregated the BGP routing table into thousands of more-specific prefixes and announced them to the Internet. Because BGP prefers more-specific prefixes than less-specific ones, most BGP routers around the world selected that ISP as the best path to

forward their data packets, while the ISP had no ability to route the packets out. Then a black hole was formed, which disrupted the Internet connectivity for hours [B+97].

As to malicious insider attacks, it is much harder to detect and prevent them because the insider attacker possesses correct identification, access control privilege, or cryptographic keys, it can easily pass identity authentication, message integrity checking and respond to any security requests correctly. The traditional approach, such as passwd-based or key-based checking may not be effective to detect such attackers.

2.2.3 Costs

The possible costs of threats are summarized as the following:

- *Traffic non-delivery:* The basic purpose of a routing system is to calculate a route to deliver data traffic to the intended destination. However, certain threats could result in bad routes and consequently the data traffic could be delivered to a wrong place where they were dropped, or a legitimate network could not receive any data, or the data delivery path contains a loop.

- *Policy violation:* The policy defines the expected path of data delivery. Certain threats could cause the deviation of such expectations. For example, a service provider might be cheated to provide transit service for a non-customer or non-peer network, or the data was delivered along a disfavored path or through a malicious network.

- *Instability:* It is desirable to have a stable path for data flows. However, due to certain threats, BGP might fluctuate between a set of paths very frequently when the network topology is unchanged, or BGP might explore a number of transient paths in a short period, or the BGP session might flap frequently.

2.3 Related Work

Kent [KLS00] proposed to use Public Key Infrastructure (PKI) to verify route announcements. Basically, this approach, well known as Secure BGP (S-BGP), assumes two PKI hierarchies which correspond to the practical assignment of IP prefixes and AS numbers. In the real world, the highest authority to assign a prefix or an AS number is the Internet Corporation of Assigned Names and Numbers (ICANN)[2]. Thus ICANN is the top node of both hierarchies. ICANN divides the whole IP space into several large IP blocks, and allocates those IP blocks to several regional authorities, such as Asia Pacific Network Information Center (APNIC), American Registry for Internet Numbers (ARIN), and Réseaux IP Européens (RIPE). Normally, each IP block in turn is subdivided into smaller IP prefixes, which will be allocated to different organizations. The AS number allocation follows the same pattern. Based on this, S-BGP requires that when a node X allocates a prefix or an AS number to another node Y, X must digitally sign a certificate associated with the allocated subject, and

[2]ICANN is formerly known as the Internet Assigned Numbers Authority (IANA)

send the certificate to Y also. When Y originates a prefix, Y must show the certificate as an evidence that it is entitled to originate the prefix. When Y prepends its own AS number in an AS path, Y must show the certificate for the ownership of the AS number as well. By this approach, one is enabled to check whether an origin AS has authority to originate a particular prefix, whether an AS path is constructed legitimately, etc.

However, S-BGP has some limitations. For example, this approach doesn't provide a way to verify the peering relationship between two ASes. One could utilize this weakness to announce false peering relationship with others. For example, when AS X receives a path $(ABCD)$, X could eliminate first two ASes, and announce a false path (XCD) without being detected. As suggested by Murphy [Mur02b], S-BGP could be enhanced by adding the information to indicate the intended receivers. In the above example, C will indicate that the intended receiver is B; thus if X announces the path (XCD), the inconsistency will be detected. In addition, S-BGP may prevent some useful aggregations, as illustrated in Figure 2.3.

In Figure 2.3, AS 2 has a certificate for the IP prefix 1.1.1.0/24. When AS 2 further allocates AS 3 with the IP prefix 1.1.1.128/25, and AS 4 with the IP prefix 1.1.1.0/25, AS 2 is able to aggregate two more specific prefixes. However, when AS 3 and AS 4 multi-home with AS 1, AS 1 cannot aggregate two prefixes because it has no certificates for the aggregated prefix. Thus AS 1 has to announce both smaller prefixes. For the same reason as multi-homing discussed in Chapter 1, AS 2 cannot perform the aggregation either, which

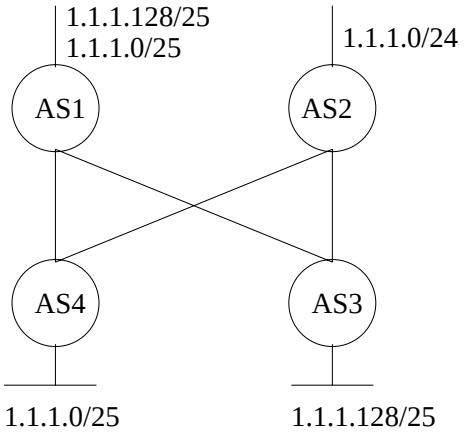


Figure 2.3: S-BGP at Aggregation Point

results in some adverse effects to the global routing system.

Most importantly, S-BGP calls for substantial modification to BGP protocol and its implementations, such as changes needed to carry the additional certificates and to query the PKI for the verification of certificates. Considering BGP has been deployed for years, it would be very difficult to realize such changes.

An alternative approach is to utilize a virtually centralize database as an authority information source to verify the routing announcements, as proposed by Yu [Yu]. The Internet Route Registry (IRR) [irr] is commonly used as such a database. IRR is designed to prompt the Internet-wide coordinations for the purpose of troubleshooting, detecting and eliminating conflicting routing requirements. IRR database contains different class of objects; one is called *Route Object*, which specifies the relationship between a prefix and an origin AS. One could utilize this information to check if an AS is entitled to announce

Object	Value
route	152.1.0.0/16
descr	NCSU
origin	AS81
mnt-by	NCREN-MAINT-MCI
changed	tas@ncren.net 19950411
source	CW

Table 2.1: A Route Object Example

a particular prefix. An example of route object is shown in Table 2.1[3]. Furthermore, RFC 2725 [VAMM99] proposed to use a strong authentication approach for delegating authority and manipulating objects in the registry.

However, this approach assumes that each AS will make their local policy public, which is not practical for the current commercial Internet. Also because it is not a mandatory operation, many ISPs are reluctant to update the entries in registries, which results in outdated and inaccurate data.

Instead of using IRR, Bates et al. [BBLR98] proposed to use DNS to store (prefix, origin AS) pairs in the originator's DNS. Each incoming route update could be checked against the DNS record to determine the correct origin AS. But given that DNS operations rely on the routing to function correctly, requiring BGP to interact with the DNS for correctness checking introduces a circular dependency. Furthermore, the DNS database can be incorrect or easily forged [AA01].

Smith et al. [SGLA96] proposed to add a signed "predecessor" to protect an AS path from being falsely modified. The predecessor of a particular AS is one of its neighboring AS. Each AS distributes its predecessor information into the

[3]The detailed information could be found in [BGJ+95]

BGP; thus the AS provides a bit of link state information at AS level. An AS could collect all of those "links", and use them to verify if an received AS path is well constructed. Path finding techniques such as those found in [GLAM95] can be used to authenticate the path.

This approach only provides a way to validate the adjacency information for an AS path; however, it is still impossible to tell if an AS path as a whole is valid or not. One could use valid "links" to make an path which violates the local policies. Moreover, it becomes more difficult to use this approach at an aggregation point because when aggregating a set of more specific prefixes into a less specific prefix, it would be very difficult to preserve the existing AS path information for those more specific prefixes. If we keep all of the AS paths along with the new less specific prefix, the BGP message could become very complex and oversize. If we drop all of the AS paths, the loss of information will open doors for an AS to falsely originate a prefix by just claiming the prefix is resulted from an aggregation.

2.4 Case Studies

In this section, we identify several cases discovered in our study which could threaten BGP or the Internet security.

2.4.1 Multiple Origin AS (MOAS)

A BGP route includes a list of ASes, called an AS path, followed by a set of IP address prefixes reachable through that AS path.[4] The last AS in the list is commonly referred as the **origin AS**. For example, an AS path of $(10, 20, 30)$ associated with the IP prefix d indicates that AS 10 learned the path from AS 20, AS 20 learned the path from AS 30 and AS 30 originated the route to d.

Figure 2.4 shows an example of AS 4 originating a route to IP prefix 128.9/16. Prefix 128.9/16 is directly reachable by routers in AS 4, and AS 4 advertises a BGP route to its neighboring ASes. AS X learns two possible routes to prefix 128.9/16, path $(Y, 4)$ and path $(Z, 4)$. In general, an AS may see many different paths leading to prefix 128.9/16, with all of them originating from AS 4. A packet destined for 128.9.176.20 (www.isi.edu) follows the BGP route for prefix 128.9/16 until it reaches AS 4 and then AS 4's interior routing protocol delivers the packet to host 128.9.176.20.

If an IP address prefix appears to originate from more than one AS, we call this a **Multiple Origin Autonomous System (MOAS)** case, or MOAS. More precisely, if prefix d is associated with AS paths $asp_1 = (p_1, p_2, \ldots p_n)$ and $asp_2 = (q_1, q_2, \ldots q_m)$, then we say a MOAS occurs if $p_n \neq q_m$.

A MOAS can be either **valid** or **invalid**. A MOAS is valid if each originating AS can directly reach the prefix. For example, Figure 2.5 shows a valid MOAS. If any of the origin ASes cannot reach the prefix, then we say it is an invalid MOAS. Figure 2.6 shows an example of an invalid MOAS involving AS 4 and

[4] In the case of route aggregation, an element in the AS path may include a set of ASes.

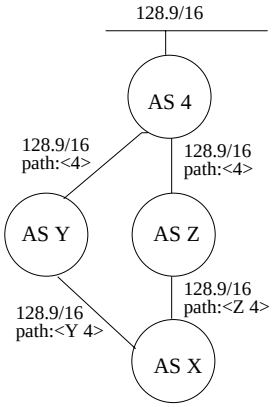


Figure 2.4: Originating a BGP Route

AS 52. Our objective is to detect invalid MOAS cases.

The problem of detecting invalid origins is a complex one due to BGP operational practices. RFC 1930 [HB96] recommends that each prefix originate from a single AS, but this is not a requirement. Legitimate operational needs, as discussed in Chapter 3, may result in a single prefix being announced by multiple Autonomous Systems. Figure 2.5 shows such an example, where the prefix 128.9/16 originates from both AS 4 and AS 226. In this case, AS X observes that 128.9/16 originates from both AS 4 and AS 226, though it has no way of telling whether this is the result of a legitimate operational need or a routing fault of the type in the following example.

Figure 2.6 shows the effect of a fault or intentional attack at AS 52. AS 52 originates a route to prefix 128.9/16 even though AS 52 cannot directly reach this prefix. With the topology in Figure 2.6, AS 52 appears to AS X to offer the shortest route to prefix 128.9/16. With today's BGP implementation, AS

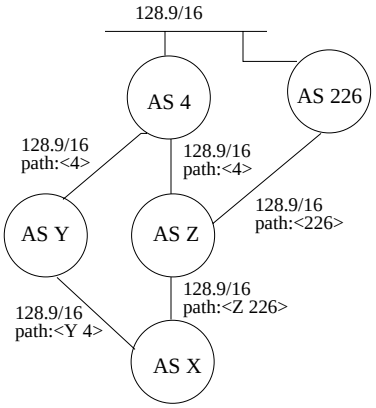


Figure 2.5: Prefix With Two Valid Origin ASes

X would accept and propagate this false route to its neighbors. Any packets destined for 128.9.176.20 that follow this faulty route would be forwarded to AS 52 instead of reaching the intended destination.

Such false route announcements have been observed a number of times over the last few years. For example, in April 2001, an operational fault caused one AS to announce routes for 9177 IP prefixes that were not reachable from that AS. These false route announcements were propagated to other ASes and packets following these bogus routes were dropped.

In Chapter 3, we present a solution based on routing information consistency checking and its full evaluation.

2.4.2 Origin AS Change (OASC)

Potentially, any AS could mistakenly or maliciously originate a prefix which is owned by others, such as MOAS, or originate a more-specific prefix implicitly

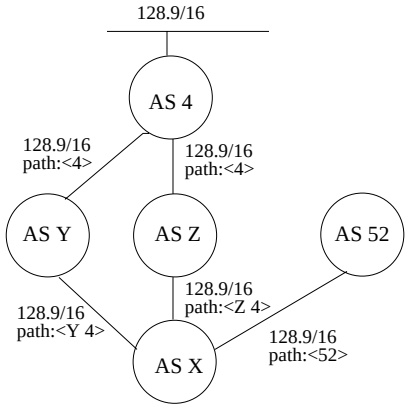


Figure 2.6: An Incorrect Origin AS

owned by others. Because BGP prefers the more-specific prefixes, the latter case may be more dangerous in this sense. OASC study concerns such problems. First, we classify OASC into different types.

Given a BGP routing table R, we only consider two elements: a prefix p and its origin AS, or ASes in MOAS cases. We can setup a mapping f between these elements, i.e., $f(p) = \{N_1, N_2, ..., N_m\}$, which means that the prefix p is originated from AS $N_1, N_2, ..., N_m$. In particular, if $f(p) = 0$, it means that the Internet Assigned Numbers Authority (IANA) owns p. In practice, IANA officially "owns" all of the IP space, and it is the authority to allocate network prefixes to different organizations; for example, 3.0.0.0/8 was allocated to General Electric Company, 18.0.0.0/8 was allocated to MIT. An organization can further allocate more-specific prefixes to others.

However, the function f doesn't consider the aggregation relation between different prefixes. In fact, if p is owned by AS_i, all of p's subnets are also owned

by AS_i except those have been allocated to others by AS_i. We use $\uparrow$ to denote the aggregation relationship between two prefixes. More precisely,

$$p_1 \uparrow p_2 \iff p_1 \text{ can be aggregated to } p_2$$

For example, 152.14.9.0/24 $\uparrow$ 152.14.0.0/16. For a set $S(p) = \{p_1, p_2, ..., p_n\}$, we define

$$p \Uparrow_S p_i \iff p \uparrow p_i \text{ and } p_i \in S(p) \text{ and } \nexists p_j \in S(p), p_j \uparrow p_i$$

Basically, p_i is the closest aggregated form for p in $S(p)$. In particular, we define $p \Uparrow_S \emptyset \iff \forall p_j \in S(p), p \not\uparrow p_j$.

A BGP routing table and its associated mapping f at time t is denoted by R_t and f_t respectively, and for simplicity, it only contains the prefixes but not their associated attributes. Given two BGP routing tables R_{t1} and R_{t2}, $t1 < t2$, we can enumerate all possible origin AS changes from R_{t1} to R_{t2} as the following:

- *C-type:* $p \in R_{t1}, p \in R_{t2}, f_{t1}(p) \neq f_{t2}(p)$. (An AS announces a prefix previously owned by another AS.)
- *H-type:* $p \in R_{t1}, p' \in R_{t2}, p' \notin R_{t_1}, p' \Uparrow_{R_{t1}} p, f_{t1}(p) \neq f_{t2}(p')$. (An AS announces a more- specific prefix out of a larger block owned by another AS.)
- *B-type:* $p \in R_{t1}, p' \in R_{t2}, p' \notin R_{t_1}, p' \Uparrow_{R_{t1}} p, f_{t1}(p) = f_{t2}(p')$. (An AS announces a more-specific prefix out of a larger block owned by itself.).
- *O-type:* $p \notin R_{t1}, p \in R_{t2}, p \Uparrow_{R_{t1}} \emptyset, f_{t2}(p) \neq 0$. (An AS announces a prefix

previously owned by IANA.)

- *H′-type:* $p \in R_{t1}, p \notin R_{t2}, p' \in R_{t2}, p \Uparrow_{R_{t2}} p', f_{t1}(p) \neq f_{t2}(p')$. (An AS stops announcing a more-specific prefix out of a larger block owned by another AS.)

- *B′-type:* $p \in R_{t1}, p \notin R_{t_2}, p' \in R_{t2}, p \Uparrow_{R_{t2}} p', f_{t1}(p) = f_{t2}(p')$. (An AS stops announcing a more-specific prefix out of a larger block owned by itself.).

- *O′-type:* $p \in R_{t1}, p \notin R_{t2}, p \Uparrow_{R_{t2}} \infty, f_{t1}(p) \neq 0$. (An AS stop announcing a prefix previously owned by IANA.)

When considering MOAS cases, each type above can be further classified by whether they involve Single Origin AS (SOAS) or MOAS. We only consider such further classification for C-type and O-type in our experiments.

- CSS: C-type change from SOAS to SOAS
- CSM: C-type change from SOAS to MOAS
- CMS: C-type change from MOAS to SOAS
- CMM: C-type change from MOAS to MOAS
- OS: O-type change involving SOAS
- OM: O-type change involving MOAS

Based on this classification, Chapter 3 presents an information visualization based anomaly detection approach to detect OASC anomalies.

2.4.3 Slow Convergence

Labovitz et al. [LABJ00] found that the delay in Internet inter-domain path failover now averages 3 minutes, and some non-trivial percentage of failovers trigger routing table oscillations lasting up to 15 minutes. Such a delay in route convergence will cause packet drops, loss of connectivity, and long end-to-end delay in the Internet. It is desirable to reduce the BGP route convergence time and minimize the number of route changes that occur during the convergence period.

Figure 2.2 shows an example of BGP slow convergence. This example, taken from [LABJ01], actually occurred in the Internet and shows one router's view of the convergence problem. Figure 2.2 shows the BGP updates sent from AS 2117 for route to a destination within AS 2129. A single BGP withdrawal from AS 2129 triggers AS 2117's six unnecessary announcements and one withdrawal. This illustrates the delayed convergence problem that occurs after a route failure, but similar problems can occur when an AS switches to an alternate route.

Note that in Figure 2.2, AS 2129 reports that it has lost its route to the destination. AS 2117 finds and announces 6 different routes to the destination, but all these six routes end in AS 2129. Since AS 2129 has lost its route to the destination, all these are routes are invalid and are eventually discarded. This illustrates the delayed convergence problem that occurs after a route failure; similar problems can also occur when an AS switches to an alternate route, i.e., *route failover*.

In Chapter 4, we propose a simple assertion-based approach to speed up

TIME	BGP Message/Event
10:40:30	Route Fails/Withdrawn by **AS2129**
10:41:08	2117 announce 5696 **2129**
10:41:32	2117 announce 1 5696 **2129**
10:41:50	2117 announce 2041 3508 3508 4540 7037 1239 5696 **2129**
10:42:17	2117 announce 1 2041 3508 3508 4540 7037 1239 5696 **2129**
10:43:05	2117 announce 2041 3508 3508 4540 7037 1239 6113 5696 **2129**
10:43:35	2117 announce 1 2041 3508 3508 4540 7037 1239 6113 5696 **2129**
10:43:59	2117 sends withdraw

Table 2.2: Slow Convergence in the Internet

BGP convergence.

2.4.4 Malicious/Unstable Routing Path

An attacker could announce a fake path which appears to be better than other routes to attract some data traffic; thus, the attacker gains the unauthorized control to those data traffic. Such an attack is commonly referred to as a route spoof attack. For example, as shown in Figure 2.7, the legitimate data traffic follows the path from client to router R1, R2, R5, then to the legitimate server. If R3 is faulty, it could advertise a fake route which appears better than (R1, R2, R5); for example, R3 could announce a direct connection with the legitimate server. As a result, R1 may select R3 as the next hop to deliver data traffic and R3 forwards traffic to some malicious machines where the data traffic might be dropped or manipulated in an arbitrary manner.

When servers considered in the above example are root or global Top Level (gTLD) Domain Name Servers, the route spoof attack may cause catastrophic

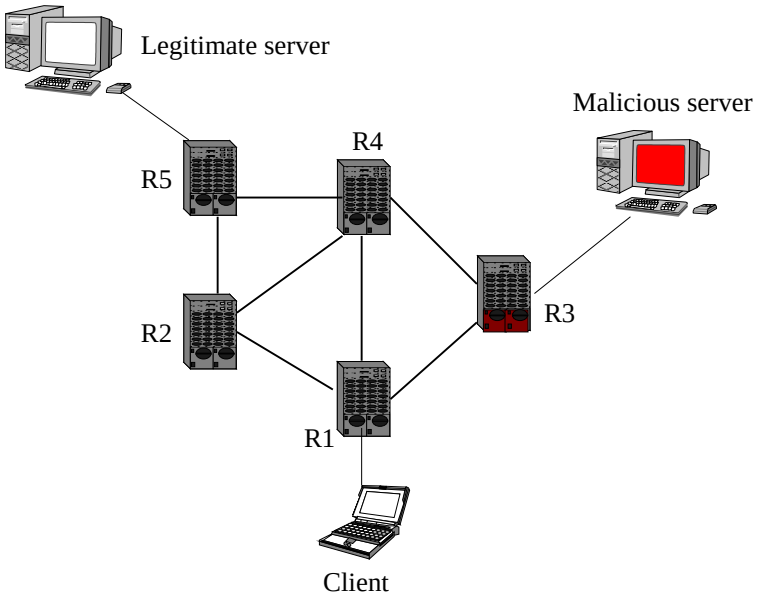


Figure 2.7: Route Spoof Attack Example

failure of Internet services because DNS is used by most Internet applications today. In Chapter 5, we present a filtering based solution, and we also fully evaluate our approach there.

Chapter 3

MOAS/OASC

As introduced in Chapter 2, MOAS and abnormal OASC could be potential threats to BGP security. In this chapter, we will focus on both problems and their solutions.

The remainder of the chapter is organized as follows. Section 3.1 presents MOAS measurement data and discusses the potential causes for MOAS cases. Section 3.2 describes our approach to detecting invalid MOAS cases and presents the evaluation results. Section 3.3 describes the information visualization based detection of anomaly OASC, followed by a summary in Section 3.4.

3.1 MOAS Measurement and Analysis

Routing data collected from the Internet operations show that MOAS does exist in today's Internet. This section presents our MOAS measurement results and

discusses the potential causes.

3.1.1 Measurement Method

We primarily used data from the Oregon Route Views Server [rou] to obtain the BGP routes and AS paths used in this study. Currently, the Oregon Route Views server peers with 54 BGP routers in 43 different ASes. Each peer exports its BGP routing table to the Route Views server.

The Oregon Route Views Server is particularly attractive because it provides data from a number of different vantage points. The data obtained from a particular local point, such as in an individual ISP, may show a smaller number of MOAS cases since fewer potential AS paths may be visible at that point in the network. For example, at a randomly selected time, the Oregon Route Views server observed 1364 MOAS cases, but three other individual ISPs observed 30, 12, and 228 MOAS cases during the same period. This only means that fewer MOAS cases were visible to these ISPs and even the number of MOAS cases observed from the Oregon Route Views Server may underestimate the total number of MOAS cases.

To obtain a relatively complete view, we used archived Oregon Route Views data from both NLANR[nla] and PCH.net [pch]. NLANR archived the Oregon Route Views data on a daily basis from 11/08/1998 to 03/16/2001. PCH.net archived the Oregon Route Views data on a daily basis from 03/16/2001 to the present. The MOAS cases are identified by prefixes only, no matter whether a MOAS case was conflicted by same set of origin ASes or the conflict was

Year	Median of MOAS cases	Increasing rate
1998	683	
1999	810.5	18.7%
2000	951	17.3%
2001	1294	36.1%

Table 3.1: Median of MOAS cases per year

continuous.

Note that AS sets did not play any meaningful role in our study. An AS path typically consists of the sequences of AS numbers used to reach prefix, but due to factors such as aggregation, the AS path may also contain AS sets as well as AS sequences. Out of over 100K prefixes observed, roughly 12 routes ended in AS sets, and these 12 routes were not included in the study.

3.1.2 Measurement Results

The total number and durations of MOAS cases deviated substantially from our expectations. Based on these results, we believe the nature of MOAS differs from what one might expect based on documents such as [HB96].

Total Number of MOAS Cases

Figure 3.1 shows the total number of MOAS cases from 11/08/1997 to 07/18/2001 [1]. Overall 38225 MOAS were observed over 1279 days. The median number of MOAS for each year is listed in Figure 3.1. There is an increase from 683 MOAS in 1998 to 1294 in 2001.

[1] The number of MOAS reached its peaks of 11842 on 04/07/1998 and 10226 on 04/06/2001.

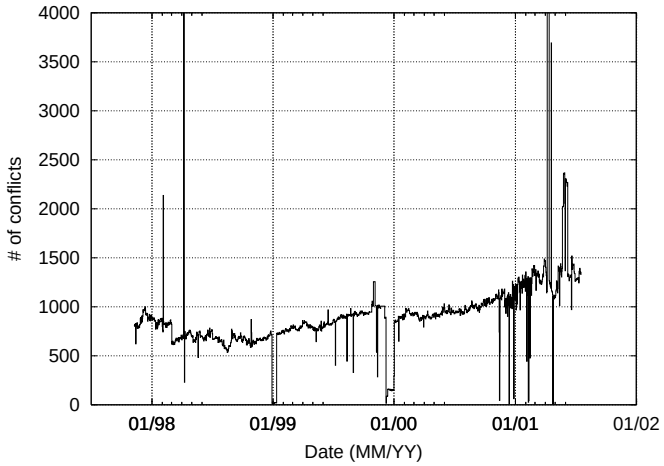


Figure 3.1: The number of MOAS cases from 11/1997 to 07/200

Duration of MOAS Cases

Figure 3.2 shows the histogram of the duration time for all the observed MOAS cases. The duration of an individual MOAS case counts the total number of days when the routes to an address prefix were announced by more than one origin, regardless of whether the days were continuous and regardless of whether the same set of origins was involved. Figure 3.2 shows that, although some small number of MOAS cases are long lasting, most MOAS cases are short-lived, for example, 13730 (35.9%) out of the total 38225 MOAS cases lasted only one day[2]. Excluding those one-time MOAS, the expectation of the duration is 30.9 days. Taking into account that many other short-lived MOAS might also be

[2]Because the Oregon Route Views Server, from which we collected our data, takes only daily routing table dumps, it is impossible for us to distinguish a MOAS case that lasts for a short time period around the time when the routing table is dumped, from one that lasts longer than one day but not long enough to appear in two consecutive routing table dumps; both cases will be considered as a one-day MOAS in our measurement.

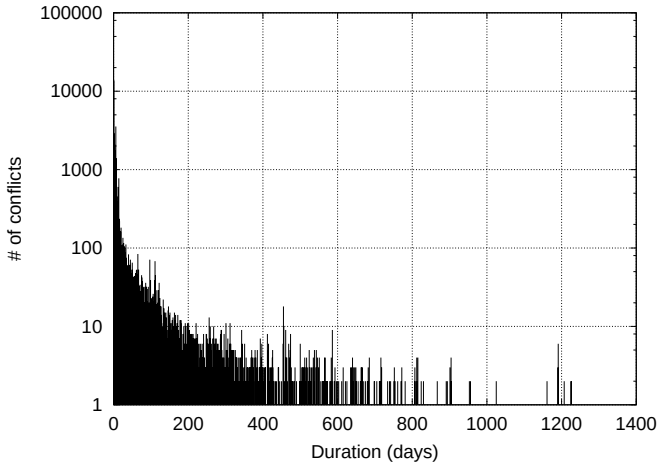


Figure 3.2: Duration of MOAS cases

Expectation (days)	Measured data set
30.9	longer than 0 day
47.7	longer than 1 days
107.5	longer than 9 days
175.3	longer than 29 days
281.8	longer than 89 days

Table 3.2: Expectation of the duration of MOAS cases

due to faults (more details given in Section 3.1.5), we considered the data set which contains only MOAS whose duration is greater than 9 days (a total of 10177 MOAS). For these MOAS, the expectation of the duration is 107.5 days with 1002 MOAS lasting longer than 300 days. Figure 3.2 lists the expectation of the duration from the different data sets. The longest duration was 1246 days out of a possible 1279 days, and 1326 MOAS were still ongoing when the measurement accomplished (Aug. 2001).

The results seem a little surprising if one assumes that multi-homing, dis-

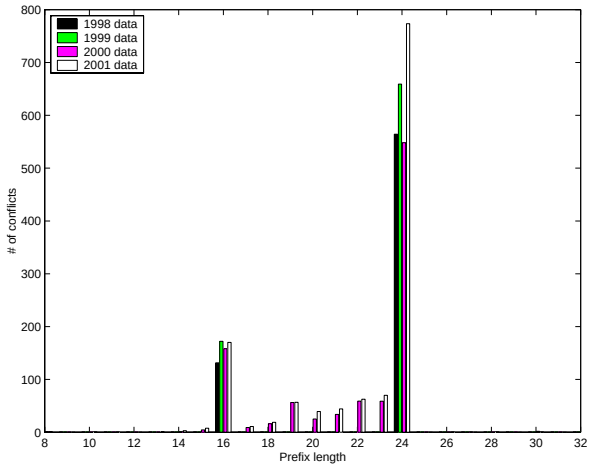


Figure 3.3: Distribution among prefix length

cussed in Section 3.1.4, is the major reason for the MOAS cases. Multi-homing would seem to imply that the MOAS cases should last longer than what is observed here, and this is discussed further in Section 3.1.5.

Distribution of MOAS Cases

Figure 3.3 shows the distribution of MOAS among prefix length. The /24 (netmask of 255.255.255.0) attracts most of MOAS. This is not unexpected since /24 prefixes make up the bulk of the BGP routing table.

3.1.3 Classification of MOAS cases

If an AS x observes a MOAS case, x will see at least two different AS paths for the same prefix d:

- $asp_1 = (p_1, p_2, \ldots, p_n)$

- $asp_2 = (q_1, q_2, \ldots, q_m)$

By definition, $p_n \neq q_m$ for a MOAS cases. In order to better understand the type of MOAS and the potential causes, we divided the MOAS cases into three classes based on relationships between the two AS paths.

OrigTranAS: $p_n = q_j$ $(j < m)$.

In this case, AS p_n announces itself as the origin AS in asp_1 and announces itself as a transit AS in asp_2.

SplitView: $p_i = q_j$ and $p_{i-1} \neq q_{j-1}$ $(i < n, j < m)$[3].

In this case, AS $p_i(q_j)$ announces different routes to different neighbors, i.e., p_i announces the path $(p_{i+1}, p_{i+2}, \ldots, p_n)$ to the peer p_{i-1}, while p_i announces the path $(q_{j+1}, q_{j+2}, \ldots, q_m)$ to another peer q_{j-1}.

DistinctPaths: $p_i \neq q_j$ $(\forall i \in [1..n], j \in [1..m])$.

In this case, there are two totally different routes for the prefix d from AS x's point of view.

Instances of all three cases were observed, and Figure 3.4 shows the number of MOAS for each class. In the OrigTranAS class, an AS acts as both the origin AS and a transit AS. In the SplitView class, a transit AS offers two different paths to the prefix and these paths end in different origin ASes.

The OrigTranAS and SplitView MOAS indicate that a single AS may advertise multiple paths to the same prefix. This is often because of the traffic

[3] $p_0 = x$ and $q_0 = x$.

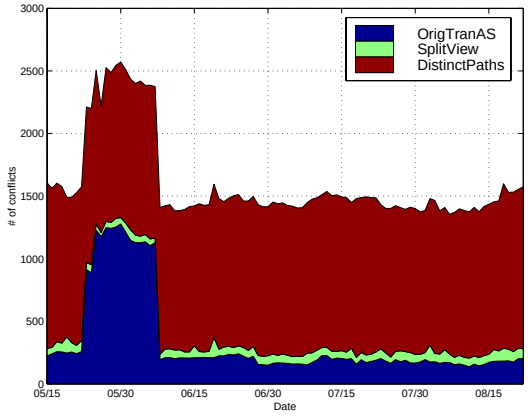


Figure 3.4: Distribution of MOAS Classes

engineering practices used at large ISPs. An AS might prefer that traffic to the same destination flow through different paths due to constraints such as geographical distances, link speed, or economic reasons.

In the DistinctPaths class, there are two completely disjointed AS paths for the same prefix. Figure 3.4 shows that the DistinctPaths class is dominant in the MOAS cases, which is not unexpected because BGP only chooses one best route if no traffic engineering is practiced.

3.1.4 Explanations and Implications of MOAS

There are a number of possible explanations for MOAS cases. Unique cases such as exchange points, some forms of multi-homing, and faults all contribute to the MOAS cases. Each of these factors was observed in this study.

Exchange Point Addresses

One potential cause of MOAS cases involves the prefixes associated with exchange points (or equivalently, links connecting ASes). A prefix associated with an exchange point is directly reachable from all the ASes at the exchange point and each AS at the exchange point might advertise the prefix as if it comes directly from that AS.

However, exchange point prefixes make up a small percentage of the MOAS cases observed in this study. In the examined BGP data, 30 out of 38225 prefixes could be definitively identified as exchange point prefixes. Our analysis of exchange point prefixes may underestimate the total number of exchange point prefixes, but the number of exchange point prefixes remains relatively small even if our estimate is off by two orders of magnitude. All of these exchange point prefix MOAS lasted for long periods, consisting of most or all of the observation periods. These MOAS cases do not present a problem for packet forwarding since each AS originating the route can directly reach the prefix.

Multi-homing Without BGP

In some cases, multi-homing can occur without the use of BGP and this can result in MOAS cases. Suppose there is a link between two ASes, but the routing across this link does not use BGP (and instead relies on static routing or some IGP). From a BGP perspective, it appears as if one AS can directly reach prefixes belonging to the other AS.

Again one would expect these MOAS to be long lasting since static routes

are likely to have a long lifetime. These MOAS cases could present a problem for packet forwarding if the links necessary to support the static routes fail.

Multi-homing with Private AS Numbers

To prevent AS number exhaustion, Haas [Haa01] suggests that a multi-homed customer use a private AS number which is mutually agreeable to all providers. This technique is called AS number Substitution on Egress (ASE). If deployed, this approach could produce MOAS cases because the private AS number should be stripped off by the upstream providers and the real origin information will be lost.

Based on discussions with network operators, we do not believe this technique is used widely in practice. These MOAS cases would not present a problem for packet forwarding since all upstream providers can reach the private AS. Furthermore, if the link to the private AS is lost, the corresponding BGP route will also be withdrawn.

Because the links using non-BGP routing mechanisms or private AS numbers are "hidden" to BGP, the pure BGP data cannot tell whether or not a MOAS case is due to multi-homing without BGP or multi-homing with a private AS number. However, by contacting individual ASes, we did confirm such occurrences.

Theoretical Causes

Other factors have the potential to cause MOAS cases, but these factors did not occur during our study. In particular, RFC 1930[HB96] notes that aggregation could result in routes that end in AS sets. But overall, we typically observed 12 prefixes which ended in AS sets, and these AS sets were consistent with each other.

Anycast address would also create MOAS cases since an anycast prefix is intended to originate from multiple ASes. No prefixes in our study were identified as anycast addresses.

Faulty or Malicious Configurations

MOAS cases can also occur when an AS incorrectly originates routes to some other organization's prefixes. This could occur due to configuration errors or even intentional attacks. Often, the faulty AS does not have a route to the incorrectly originated prefixes, and packets that use the incorrectly originated route will reach the faulty AS and then be lost.

Figure 3.1 shows several notable examples of MOAS cases caused by faults. The graph shows a large spike on April 7th, 1998, and AS 8584 was involved in 11357 out of 11842 MOAS that occurred during that day. Discussions on a network operators mailing list[nan98a] indicated that AS 8584 falsely originated routes to those prefixes. Consequently, some ASes selected the incorrectly originated route. Packets sent along this incorrectly originated route would reach AS 8584 and would then be lost.

The graph also shows a large spike on April 10th, 2001 and the sequence (AS 3561, AS 15412) was involved in 5532 out of 6627 MOAS cases that occurred during that day. Based on the archived data from RIPE RIS [ris], AS 15412 normally originates only 5 prefixes. However, on April 6th, AS 15412 suddenly originated thousands of prefixes due to a configuration error[nan01].

On April 25th, 1997, a severe Internet outage occurred[B+97] when one ISP falsely de-aggregated most of the Internet routing table and advertised the prefixes as if they originated from the faulty ISP[nan97]. The falsely originated prefixes resulted in MOAS. These examples show that invalid MOAS cases do occur and can have serious effects on Internet routing.

Faulty aggregation could also cause MOAS cases. In faulty aggregation, an AS advertises an aggregated prefix, even though some of more specific prefixes are not reachable by the AS. A MOAS case occurs if an aggregate route is also generated by some other AS. Packets that use the faulty aggregated route will travel to the faulty AS and then may not be able to reach all the more specific prefixes.

3.1.5 Re-examine MOAS Durations

With the exception of faults and intentional attacks, the possible explanations should have created long duration MOAS cases. MOAS cases for exchange point prefixes should remain as long as two or more ASes choose to advertise a route to the exchange point. The data confirmed this expected pattern and exchange point MOAS cases persisted for most, if not all, of the study. Multi-homing

without BGP and multi-homing with Private AS numbers both require router policy configurations at two or more ASes, and the resulting MOAS should persist for as long as the multi-homing policy remains in place. We expected that multi-homing policies (and the resulting MOAS cases) would occur over months, not days. But the data in Section 3.1.2 show a large number of short duration MOAS.

One possible reason for short-lived MOAS cases is that MOAS cases could occur during a transition period when a non-BGP customer switches from one provider to another. To guarantee the connectivity to the non-BGP customer, it is possible for both providers to originate the customer's prefix for a short period. Another possible reason for short-lived MOAS cases is router mis-configurations or other faults. These MOAS disappear when the faults are detected and corrected. Furthermore, those extremely short-lived MOAS cases, with a duration of one or two days, may suggest an unintended behavior. In fact, 13730 (35.9%) out of the total 38225 MOAS cases lasted only one day, and 82.7% of these short-lived MOAS cases can be attributed to a configuration fault that occurred on April 7th, 1998.

Overall, the duration can be a useful heuristic to distinguish between valid MOAS cases and invalid ones. However, such differentiation cannot be accurate enough to be a solution to validate MOAS cases.

3.2 Detecting Invalid MOAS

Blind acceptance of MOAS that occurs in BGP announcements is dangerous because invalid MOAS cases could adversely affect packet forwarding. In this section we describe a simple mechanism that allows BGP routers to distinguish invalid MOAS cases from valid ones.

3.2.1 MOAS List

Our solution is to first create a list of the multiple ASes who are entitled to originate a particular IP address prefix p, and then attach this list to the route announcements by all those originating ASes. BGP routers that receive the route announcements from multiple origins can verify that the MOAS is intentional and valid. If another AS makes a faulty route announcement to prefix p, BGP routers which have received the right route to p can easily detect the fault since this faulty route's origin AS will not be in p's MOAS list. When a conflict between different MOAS lists for the same prefix p is detected, an operational alarm could be raised to alert human operators or an automatic query to an authoritative database could be issued to find out who is qualified to originate p.

For example, suppose multi-homing allows prefix p to be originated by both AS 1 and AS 2. A MOAS list will be attached to the routing announcements indicating that both AS 1 and AS 2 can serve as the origin AS for this prefix. A faulty AS, say AS 3, may also originate a route to prefix p, but AS 3 does not appear in the MOAS list advertised by AS 1 and AS 2. Although AS 3 could

attach its own MOAS list that includes AS 1, AS 2, and AS 3, this list would not be in agreement with the MOAS list advertised by AS 1 and AS 2. Any router that sees both the faulty route and at least one of the valid routes can compare the MOAS lists and detect that there is potentially a problem.

However, if the origin AS(es) for p has only one path to reach the rest of the Internet, a fault or attack can defeat the MOAS detection mechanism by altering the origin AS or the origin AS list on this single path. But in this case, the attacker has compromised the only path to reach p and can cause other arbitrary damage to p as well. In more general cases, multiple origin ASes make route announcements for p and/or the origin AS(es) announces its route to multiple AS peers. As we demonstrate in our simulation, it is difficult for attackers to block or modify the origin AS list on all of these route advertisements, especially considering the increasing inter-connectivity in today's Internet topology [Hus01a].

The origin MOAS list does not use cryptographic authentication and may be removed or altered, either intentionally or unintentionally, as the route propagates through chains of ASes. Our technique relies on the distributed nature of the Internet topology for fault detection. While it may be possible to tamper with the routes to the prefix p along some of its propagation paths, trying to tamper with the routes to p along all the paths that the route p announcement propagates would seem very difficult, if not impossible, in a large, well connected network topology. As long as the correct route p announcement can propagate out to a number of other ASes, it is likely that the conflict due to

the tampering will be detected, thus protecting the routing system from blindly accepting bogus routes injected by potential attacks or faults.

3.2.2 Implementing the MOAS List in BGP

The BGP community attribute [CTL96a] provides a simple way of attaching the MOAS list to a route announcement. The community attribute is an optional transitive BGP attribute of variable length. It can be used to convey additional information to the global routing system for a group of prefixes that share some common properties. Each community attribute consists of four octets. By convention, the first two octets are used to encode an AS number and the semantics of the final two octets may be defined by the AS listed in the first two octets. We propose to reserve one of the 2^{16} values available in the last two octets to indicate a MOAS list. This value is denoted by $MLVal$, *MOAS List Value*, in the remainder of this paper. The community attribute $(X : MLVal)$ indicates that AS X may originate a route to this prefix. The MOAS community value is formally specified in [ZMM+01].

For example, if a prefix p is originated from all of AS_1, AS_2, ..., AS_n, the route updates from $AS_i, (1 \leq i \leq n)$ will include the MOAS List $(AS_1, MLVal)$, ..., $(AS_n, MLVal)$. In Figure 3.5, AS 1 and AS 2 agree that both of them may originate routes to the same prefix p. When AS 1 originates p, AS 1 will attach the MOAS List, as shown in Figure 3.6. Similarly, AS 2 will attach the same MOAS list to its route announcement for p. An example configuration file is shown in Figure 3.7.

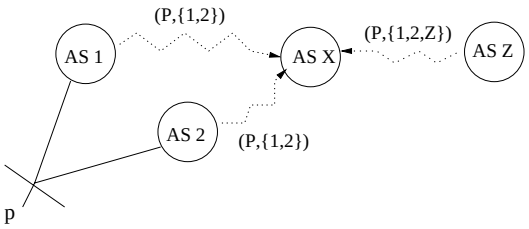


Figure 3.5: Example scenario

AS 1
$MLVal$
AS 2
$MLVal$

Figure 3.6: Example MOAS List

router bgp AS1
neighbor a remote-as Y
neighbor a send-community
neighbor a route-map setcommunity out
route-map setcommunity
match ip address p
set community AS1:$MLVal$ AS2:$MLVal$

Figure 3.7: Example configuration file

When a BGP router receives route announcements for the same prefix p from multiple origins, it checks to see whether the MOAS Lists for p from all the announcements are consistent[4]. Here, the consistency is defined as the same set of ASes listed in all the MOAS Lists. The order in the list may differ, but the set of ASes included in each route announcement must be identical. Whenever a BGP router notices any inconsistency in the MOAS Lists received, it should generate an alarm signal; further investigation should be conducted to identify the cause of the inconsistency.

In Figure 3.5, both AS 1 and AS 2 attached a MOAS List, $(AS1 : MLVal, AS2 : MLVal)$, to their route announcements for p. If AS Z falsely originates a route to p with a MOAS List $(AS1 : MLVal, AS2 : MLVal, ASZ : MLVal)$, another AS, say AS X, will observe an inconsistent MOAS List and will generate an alarm.

Attaching MOAS List to route announcement requires only BGP configuration changes. Checking MOAS List consistency, on the other hand, requires that BGP implementation be modified accordingly. However one could deploy the MOAS List checking quickly in the operational Internet via an off-line monitoring process, which periodically downloads the BGP routing messages and checks the MOAS List consistency from multiple peers. If the router is equipped to support the new BGP MIB [HHW+01], one could also run a management application to get all MOAS List through the MIB interface and check the MOAS List consistency.

[4]If a route does not contain a MOAS list, it will be treated as if it carries a MOAS list containing the origin AS.

Note also that checking the MOAS list is a single set comparison. When an route update for p is received, the MOAS list is simply compared with the existing MOAS list for p (or is simply accepted if this is the first and only announcement for p).

3.2.3 Limitations of the Proposed Solution

There are a few issues regarding a wide deployment of our solution in the Internet today. In particular, given that BGP community attribute is an optional transitive value, some routers may drop community attribute values associated with a route announcement, an allowed behavior under the current specification. When a router receives multiple route announcements to the same prefix p, some with MOAS list and some without, it would raise a false alarm. However we note that dropping the MOAS community value from some route announcements should not cause an invalid case to be considered valid, as long as such dropping is limited to a fraction of all the route announcements.

The attachment of a MOAS list also adds to the overall size of the routing table and route announcements. Routes that originate from a single AS need not attach a MOAS list. A route with no MOAS list attached implies that the route may only originate from the AS listed as the last one in the AS path. Our earlier measurement results [ZPW+01] showed that in today's Internet less than 3,000 routes originate from multiple ASes (including the routes that incorrectly originated from multiple ASes). Furthermore, about 99% of all MOAS cases involve 3 or fewer origin ASes. Thus the MOAS list itself should be relatively

short.

Like other security mechanisms, our approach might become a target of attacks. One possibility is that an attacker could invoke conflicts intentionally to prevent the correct routes from being adopted. In this case, a router could ignore MOAS list to ensure the network connectivity or it could hold all the conflicted routes for a while to avoid security risks, depending on the local policies and security requirements for a particular situation.

Our simple MOAS solution, as described in this paper, helps enhance BGP reliability by distinguishing valid MOAS cases from invalid ones. In its current form, however, it may not be effective in detecting more complex forms of invalid routing announcements. For example, an AS could make a false route announcement with a correct origin AS but a manipulated AS path, or it could falsely announce a route to a prefix longer than p where p is an IP address prefix belonging to another AS. However, our simple MOAS solution shows a first example of how one may utilize the existing network topology itself in detecting faults. We are continuing our work in this direction by enhancing the current solution to detect more complex routing faults.

3.2.4 Identifying the Correct Origin AS

With our simple MOAS solution, a route announced by a false origin will conflict with the route carrying the correct MOAS list, causing an alarm to be raised. Once an alarm is raised, the router (or network administrator) needs to distinguish the route with correct origin AS(es) from the one with the false

origin.

There exist a variety of potential approaches to determine the correct origin AS(es). One possibility is to enhance the DNS database to carry the information of valid origin AS(es) for each address prefix, as proposed in [BBLR98]. In this approach, whenever a MOAS case for prefix p occurs, the router performs a DNS lookup to verify the origin AS of p by specifying the DNS Resource Record type as $MOASRR$. If the origin AS in a route announcement does not match any AS number in the AS list of DNS $MOASRR$ record, the route announcement should be considered as bogus. DNS security [MR01, Eas99] can be used in this approach to assure the correctness of the DNS database.

However, the difference between our proposal and [BBLR98] is that by combining our solution with DNS-based checking, we minimize the frequency of DNS queries from BGP routers; only in cases of invalid MOAS or dropped MOAS lists will DNS queries be triggered, instead of invoking DNS-based checking every time a new route is announced.

There are some limitations to use DNS to validate the correct origin AS. Theoretically, using DNS service as an auxiliary function for the routing system might form a dependency loop because delivering DNS queries to DNS servers requires the correct routing first. Also, it is possible that DNS data may be out-dated or corrupted. Overall, it would be helpful to utilize DNS service as a complimentary way for routing information checking, but it may not be used in a determined way. Further research on the topic of validating origin AS is on-going.

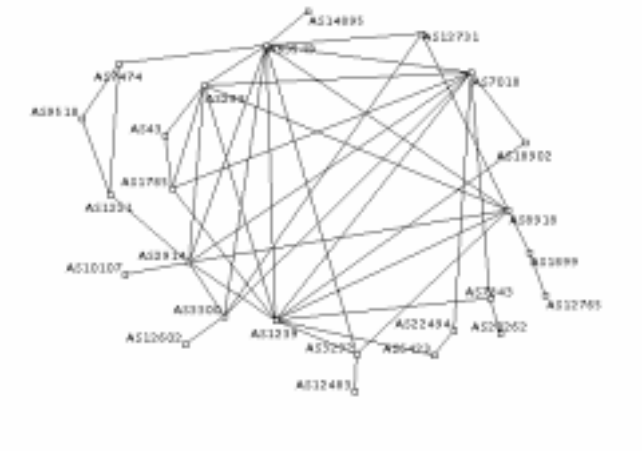
3.2.5 Simulation

We used simulation to evaluate the effectiveness of our approach. More specifically, we assume a model where attackers inject false routing announcements at randomly selected locations. We compare the damage the attackers may cause with and without our MOAS solution. We also examine the effectiveness of our solution with different topology sizes and with partial deployment. In the rest of this section we first describe the simulator and the topology used in our simulation and then present the results from three experiments.

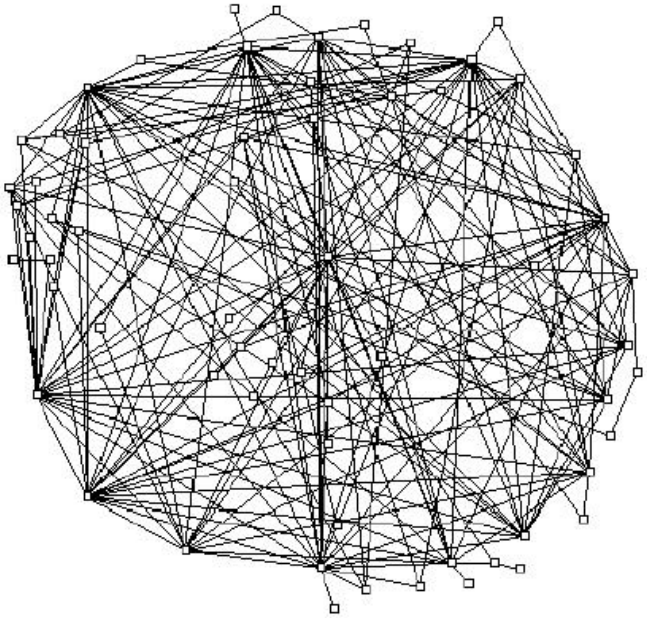
Simulation setup

We use a modified version of the BGP simulator in SSFnet [ssf] in our simulation. Our simulations use three topologies: a 25-node topology, 46-node topology, and 63-node topoology. In the simulation topologies, each node represents an Autonomous System (AS), and a link between two nodes represents a BGP peering connection (i.e., the two ASes exchange routing information). Figure 3.8 shows the 25 node and 63 node topologies. The 46-node topology is similar but is omitted for brevity.

In order to generate simulation topologies close to the actual Internet topology, we first get the full BGP routing table from the Oregon Route Views Server [rou]. Then we infer BGP peering relations based on the AS Path attribute in the collected BGP routes. For example, if a route to a prefix p has the AS Path *1239 6453 4621*, we consider AS 6453 to have two BGP peers, AS 1239 and AS 4621. We also mark AS 6453 as a *transit AS* since packets to and



(a) 25-AS Topology



(b) 63-AS Topology

Figure 3.8: Simulation Topology - 63 AS

from AS 4621 may traverse through it (note that AS 1239 is also a transit AS). If an AS does not appear to be a transit AS in any of the routes, we consider it a *stub AS*. Transit ASes represent ISPs (e.g., AS 1239 is Sprint), while stub ASes are networks at the edges of the Internet such as commercial companies and universities. Next, we randomly select $x\%$ of the stub ASes and construct a topology containing these stub ASes and their ISP peers, with the peering relations among all the selected ASes completely preserved. If a transit AS has only one peer left after the initial section, we prune it from the topology. Since the removal of an AS may make it necessary to prune its peer if that peer will have only one or no neighbors left, the pruning needs to be done iteratively. Finally we inspect the topology to make sure that it is a connected graph.

To generate MOAS, we randomly select origin ASes from the stub ASes. In our simulation, each prefix is originated by either one or two valid origin ASes. We do not simulate prefixes that are correctly originated by more than 2 origin ASes since according to our measurement, 96.14% of MOAS cases involve two ASes and 2.7% involve three ASes. We allow any number of attacker ASes to originate invalid routes to the prefix, and we choose the attacker ASes randomly from all the ASes. Note that the attackers may have a higher probability to block more valid routes if they are located in transit ASes. Stub (non-transit) ASes may have a lower level of security, but compromise of a stub AS is less valuable to an attacker since the attacker has less ability to block valid routes.

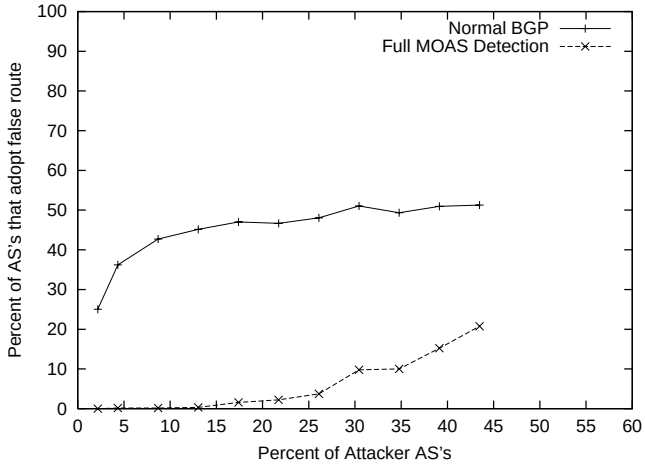
Experiment 1: Effectiveness of MOAS List

In this experiment, we evaluate how effectively our scheme prevents the propagation of false routing information, by comparing the number of routers adopting false routes with and without using the MOAS List. We assume that all the nodes check the MOAS attributes received from their peers and, once they detect a MOAS case, they stop the further propagation of a false route (e.g., by checking with DNS as proposed in the paper or using some other mechanism).

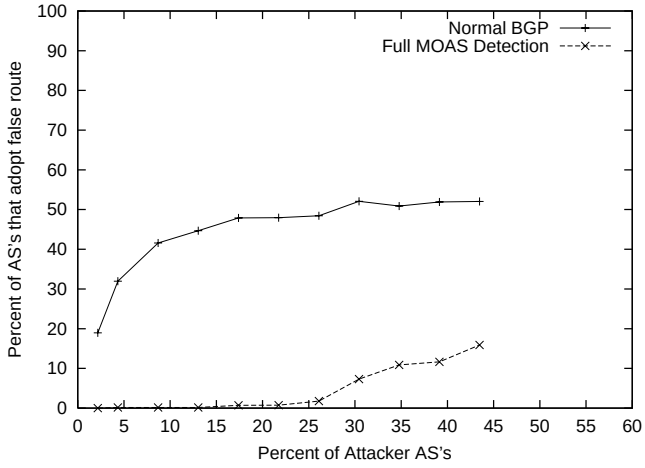
We randomly select either one or two origin ASes for a prefix and then randomly select M attacker ASes. An attacker AS will incorrectly advertise a route to the prefix. It is easy to see that the number of different selections can be rather large for large topologies. Therefore, rather than simulating all the possible selections, we perform 15 runs for a given number of origin ASes and attackers[5]. In other words, each data point is the average of 15 simulation runs.

Figure 3.9 shows the results for the 46-AS topology, the X axis is the percentage of attackers over the total number of the ASes, and the Y axis is the percentage of the remaining ASes (excluding attackers) adopting to the false routes announced by the attackers. As one can see, when the number of attackers increases, more (non-attacker) ASes are affected by the false routing information. However, deployment of our simple MOAS solution reduces the percentage of (non-attacker) ASes adopting the false routes greatly. When up to 4% of the ASes are injecting false routing data, more than 36% of the re-

[5]To get the 15 combinations of origin ASes and attackers, we first select 3 sets of origin ASes from the stub ASes. Then we select 5 sets of attackers for each set of origin ASes.



(a) 1 Origin AS



(b) 2 Origin ASes

Figure 3.9: Spoof-Resilience of Our Scheme in the 46-AS Topology

maining ASes will adopt false routes without validation of the routes. With our solution, on average only 0.15% of the ASes adopt false routes in the same simulation setting. Even when the number of attackers increases to 30% of the network, only about 9.8% of the remaining ASes adopt false routes, compared to 51% when the MOAS list is not employed. The results are similar for the 25-AS and the 63-AS topologies.

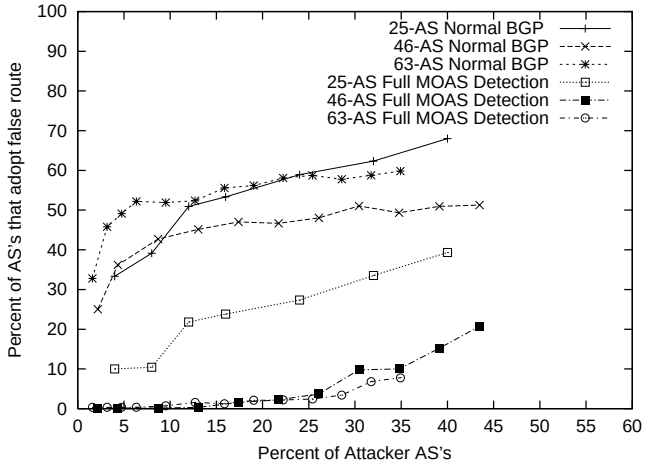
Experiment 2: Simulation using Topologies with Different Size

The goal of this experiment is to examine the effectiveness of the MOAS solution in larger topologies. We also compare the results from this experiment with those from the previous experiment to understand the impact of topology size on the robustness of our solution. The topology we use here is the network with 25, 46 and 63 nodes, respectively. We have run the experiment with both one origin AS and two origin ASes; the results are similar as shown in Figure 3.10.

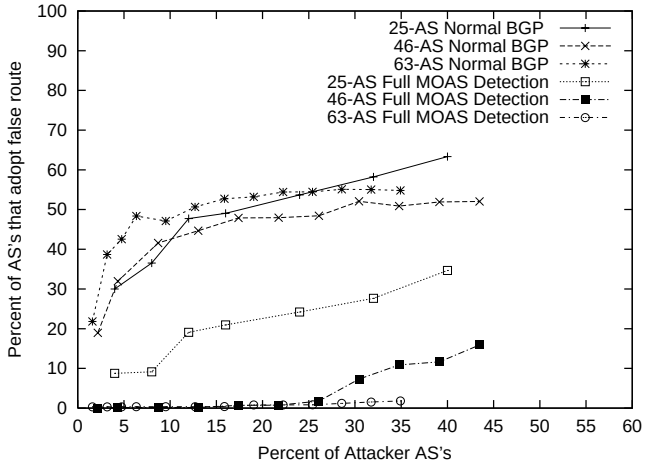
One can make the following observations from Figure 3.10(a):

1. Without our MOAS solution, the effects of the attackers on the two topologies are quite similar (the gap between the top three curves is much smaller than the gap between the other three curves).

2. With our scheme, the larger 63-AS topology is more robust against random attackers than the smaller 25-AS topology. When the attackers are less than 20% of the total number of ASes, only 2.1% of the remaining ASes are affected by the false routing information. Even when about 35% of the ASes are compromised and announce false routes, only 7.8% of the



(a) 1 Origin AS



(b) 2 Origin ASes

Figure 3.10: Comparison between 25-AS, 46-AS and 63-AS Topology

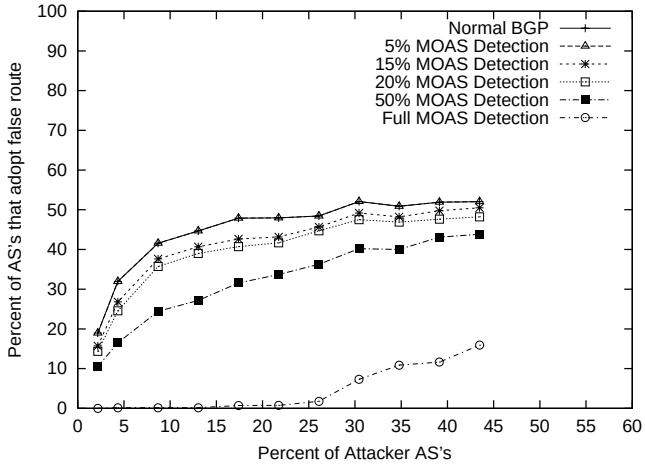
remaining ASes adopted false routes in the 63-AS topology, compared to about 31.2% of (non-attacker) ASes in the 25-AS topology.

The above results suggest that our scheme becomes more effective in larger topologies. We believe that the improved robustness of our solution comes from the fact that ASes are more richly connected in the larger topology, which enables route announcements with the correct AS or correct MOAS lists to reach more ASes. As a result, more ASes detected the inconsistency between correct routes and false routes by the attackers. As part of our continuing research effort we are currently seeking a formal validation proof of this phenomenon.

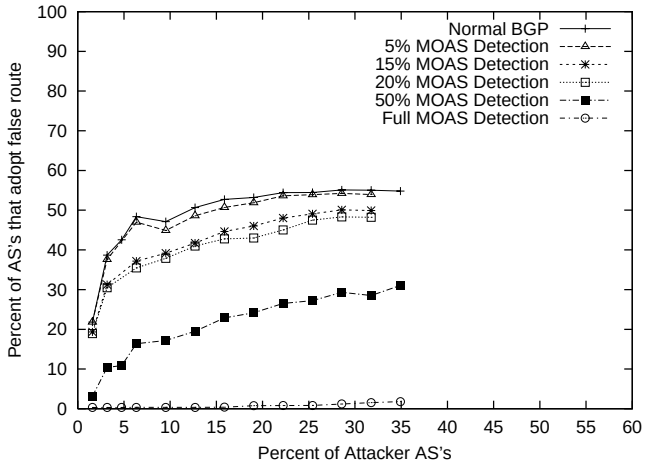
Experiment 3: Partial Deployment of MOAS Checking Capability

This experiment evaluates the effectiveness of our MOAS solution when it is only partially deployed in the network. To simulate partial deployment, we randomly select 50% of the nodes to have the capability of processing MOAS List, i.e., they can distinguish between valid and invalid MOAS cases and eliminate false routing announcements. The other nodes ignore the MOAS List, which means they may accept and install a false route in their routing table and advertise the false route to their peers if this route is considered the best path to reach a prefix.

In Figure 3.11, we compare the effect of partial and full deployments of MOAS detection. As the figure shows, even when only half of the nodes can detect MOAS cases, they can still provide protection benefit for the other nodes, because these MOAS-capable nodes stop the false routes from further propa-



(a) 46-AS Topology



(b) 63-AS Topology

Figure 3.11: Comparison between Partial and Complete Deployment of MOAS Detection

gation through them. For example, in the 63-AS topology, 50% deployment of MOAS detection reduces the percentage of ASes adopting the attackers' routes by more than 63% in the presence of 3% attackers. However, 5% deployment in a 46-AS topology has no effects at all. It is because in this case, only 2 ASes will be equipped with MOAS detection capability, and it would be very difficult to detect invalid routing announcements. One can also observe that the larger topology performs better than the smaller topology when MOAS detection is partially deployed.

3.3 Detecting OASC Anomalies

Jointly working with another student, Soon Tee Teoh, at UC Davis, we attempted to advance one step further to examine the Origin AS Changes (OASC). As introduced in Chapter 2, the prefix may change its origin AS due to economic, operational, or administrative reasons. We believe such changes should be relatively rare because a prefix normally keeps a stable relation with its origin AS, or ASes. However, due to unintentional faults or malicious attacks, abnormal changes may be observed, as shown in this work.

We used an idea similar to the anomaly detection commonly used in the Intrusion Detection Systems. In addition, some information visualization techniques were exploited, including a special encoding of IP addresses and a built-in interactive visual interface, which allows a user to recognize the anomalous OASC. We demonstrated that each visually spotted anomaly agreed with actual

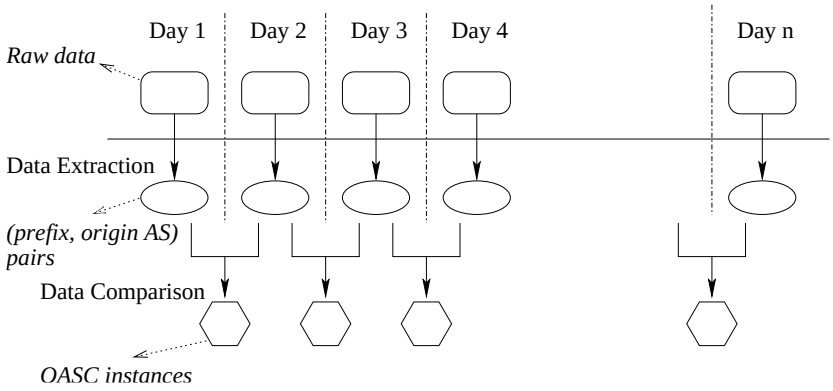


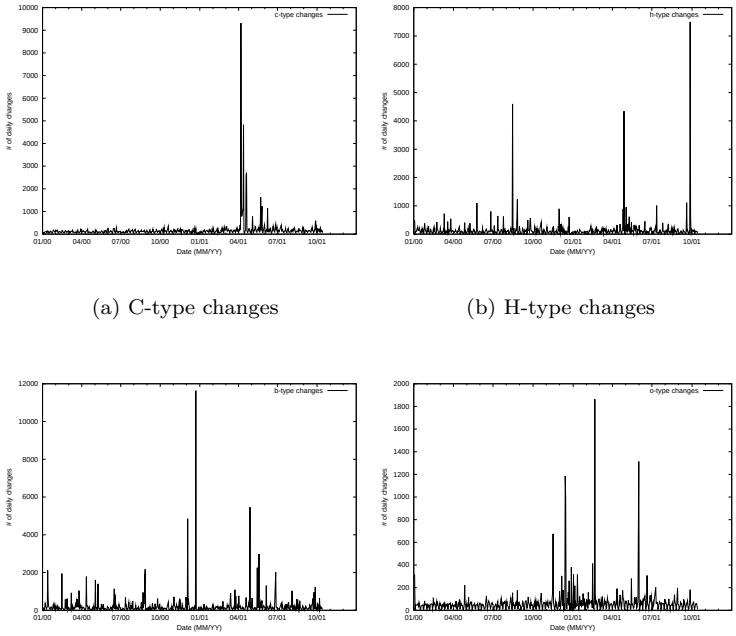
Figure 3.12: OASC Data Process

an anomaly on record.

In this work, Soon Tee Teoh mainly developed the visual-based anomaly detection system, while I collected the data, conducted the OASC analysis, and identified several anomalies by using this system.

3.3.1 OASC Measurement

We obtained the archived daily BGP routing data over 480 days from the Oregon Route Views Server [rou]. From these raw data, we extracted the prefixes and their origin ASes into a daily set of $(prefix, originAS)$ pairs. Then OASC instances are generated by comparison of two consecutive day's data. This procedure is illustrated in Figure 3.12, and some results are shown in Figure 3.13. Note that as an experiment, we only classified OASC into the first four types described in Chapter 2, namely, C,B,H,O-type changes. C-type changes and O-type changes are further classified based on the involvement of MOAS. Totally, we have 8 different OASC types in our study.



(a) C-type changes

(b) H-type changes

(c) B-type changes

(d) O-type changes

Figure 3.13: OASC Data

3.3.2 A Visual-Based Approach

Given the size and complexity of the Internet, we believe that, at least in the short term, a network system with machine intelligence alone will have certain limitations in detecting and responding to novel attacks/faults targeting on BGP. Another complementary approach to handle BGP vulnerabilities is via an interactive process between network administrators/operators and network management systems with visualized network information. For instance, given a colorful image of some BGP routing data, an experienced human operator can discover numerous facts about the Internet instantly, while an analysis program must already have the mechanisms built in to achieve the same results.

Most existing anomaly detection methods are based on statistical analysis [Lan99, LTG+92, Lun93]. That is, a set of "normal" data is first analyzed to derive representative characteristics of normal use, which are then compared against the characteristics of unknown data to disclose abnormal behaviors. This comparative analysis forms the basis of anomaly detection. Normally, it requires an anomaly detection algorithm to be trained for a relatively long time to gain the "sense" of what is normal/abnormal. Our approach does not need a "normal" data set and mainly relies on the superior visual processing capability of the human brain to detect patterns and draw inference. Starting with no prior knowledge of what shape or form the anomalies take, we use visualization as the key tool for discovering the intrinsic properties of normal and abnormal data.

In traditional statistical methods, it is a challenge to set threshold values

such that false positives are minimized while not missing true positives. With the visual approach, we relegate the responsibility for making such fuzzy judgment to the human users. Furthermore, the user can judge whether a detected anomaly is important or is just an isolated case, whereas an automatic method would just raise flags based on a rigid set of criteria. One of the most difficult tasks in intrusion detection is "event correlation"; but via human visualization, this task might be much more plausible.

There are three goals of our visualization system. The most important one is for the user to be able to quickly identify anomalies in the data. However, it is not enough merely to discover that an anomaly has occurred. Therefore, two additional goals are to enable the user to quickly understand the nature of the anomaly and to identify its source. This is so that the user can know where to focus further investigation and take corrective action. With appropriate visual metaphors, these two additional goals can be more easily achieved than with automatic, non-visual techniques. In the rest of the section, we will briefly review the work Soon Tee has done to develop such a system.

Mapping IP Prefixes

Each IP prefix maps to one pixel on a square. The mapping is done in a traditional quad-tree manner. Figure 3.14 shows this mapping. In a quad-tree, a square is repeatedly subdivided into 4 equal squares. In mapping a 32-bit prefix to a square, we start with the first two most significant bits of the address to place the IP address in one of the 4 squares in the second level of the

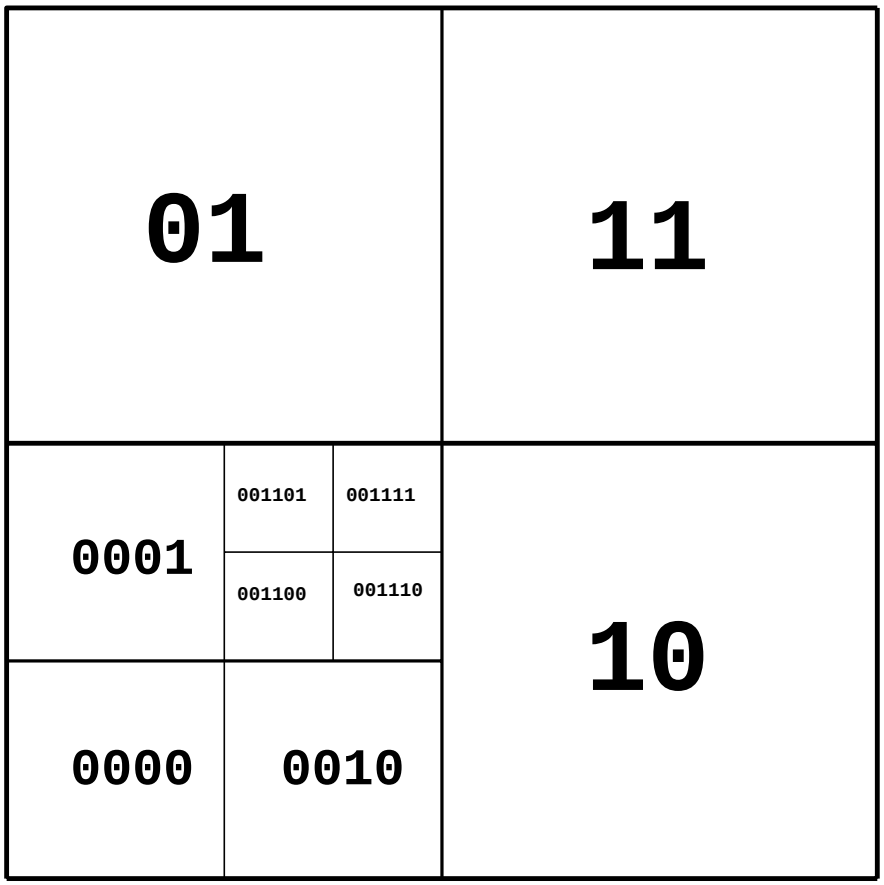


Figure 3.14: Quadtree coding of IP prefixes, show top few levels of the tree, and the most significant bits of the IP prefixes represented by each sub-tree (sub-square).

quad-tree. We then use the next two most significant bits to place the IP prefix in the appropriate third level square within this square. We do this repeatedly until we can place the prefix in a square the size of a single pixel. The prefix is mapped to that pixel.

Due to the limitations of a computer screen space, we use a 512×512 pixel square to represent the entire 32-bit IP prefix space. With only 512×512 pixels,

even though many IP prefixes map to the same pixel, we found that a 512×512 square is sufficient in spreading out the IP addresses in our data. With an additional level of zooming into a portion of the data, we can view individual IP prefixes. Figure 3.15 shows additional windows offering closeup views of several different portions of the main window, which shows the entire IP prefix space.

In the main window, a pixel is colored yellow if an OASC instance occurred on the current day, and colored brown if a change occurred on a previous day. In the detail windows, a colored square is shown for each Origin AS change. The position is determined by the IP prefix, the size by the mask, and the hue by the type of the change. Each of the 8 different possible types of Origin AS change is mapped to one unique hue. The brightness of each square depends on the day the change occurred, with the current day's data being the brightest. This example shows the data over a 416-day window from January 1, 2000 to February 19, 2001. To show only one day's data, the user can set the window to one day.

This is a sensible mapping from IP prefix to screen space because IP prefixes sharing similar most significant bits would be in close proximity on the screen. In the detail windows, each IP prefix is shown as a square or a rectangle. The size of the rectangle indicates the size of the block of IP addresses; prefixes with a smaller mask get mapped to larger rectangles.

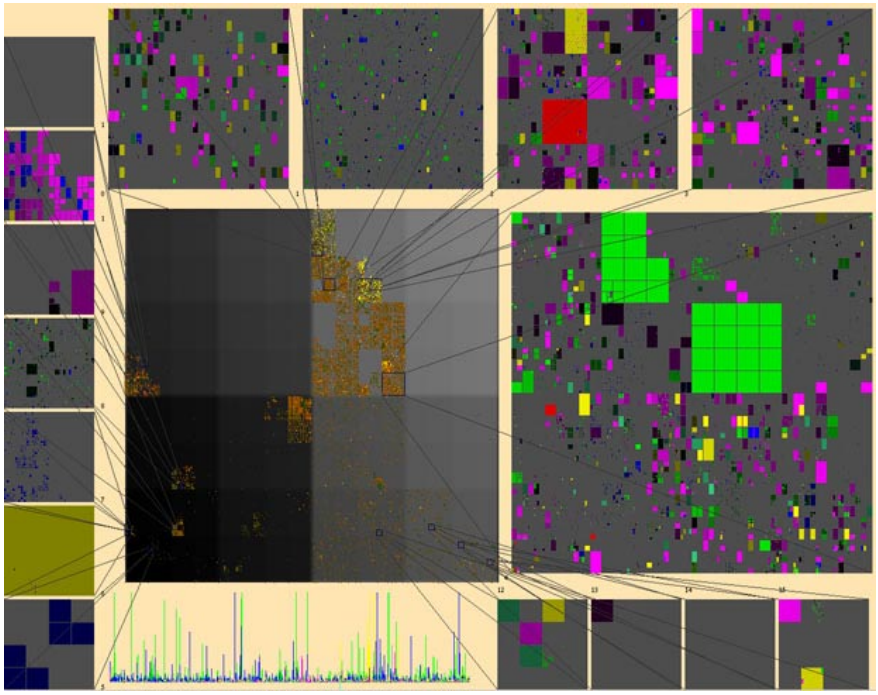


Figure 3.15: Visualization of data for 416 days up till February 19, 2001. The main window shows the quadtree mapping of the entire space of 32-bit IP address. A pixel is colored yellow if an Origin AS Change occurred on the current day (February 19, 2001), and colored brown to green if a change occurred on a previous day (January 1, 2000 through February 18, 2001). In the windows showing detail, a square is used to depict each change, with hue determined by the type of the change, brightness determined by how long ago the change occurred (present day data shown the brightest), and size determined by the mask of the prefix. The background of the main window is shaded according to the IP prefix the pixel represents. The brighter the pixel, the larger the IP prefix represented.

Relationship Between Prefix and AS

Next, the relationship between a prefix and its associated AS number needs to be represented. To achieve this, we place 4 lines surrounding the IP square, and an AS number is mapped to a pixel on one of the 4 lines. A line is then drawn from an IP address to an AS number if there is an Origin AS change involving that IP address and that AS number. This mapping takes advantage of the user's acute ability to recognize position, orientation, and length. Figure 3.16 shows the visualization of the IP-AS relationship of Origin AS Changes of a typical day. Once again, the color of each line is based on the type of change it represents.

Since there are more AS numbers than pixels, more than one AS number maps to a pixel. Again, we provide zooming features for the user to differentiate between AS numbers which map to the same pixel in the main display. The lines representing changes for the AS in focus are shown with brighter and more saturated colors than other changes. This effectively highlights the AS, fading the other changes into the background. This is shown in Figure 3.17, where the pink (OS-type) lines emanating from one AS are highlighted among thousands of lines.

Animation and Other Features

For the time dimension, our design shows one day's data at a time, and allows the user to animate the visualization, each frame showing consecutive day's data. With this "movie" display, the user can detect temporal patterns by

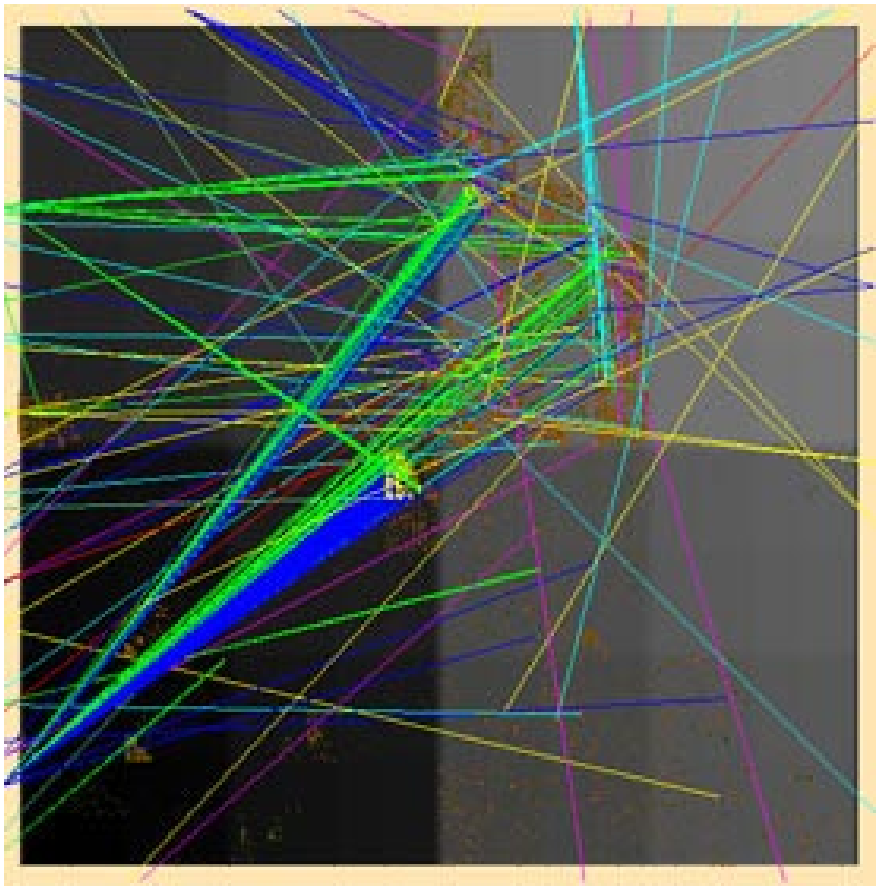


Figure 3.16: Data on a typical day (September 24, 2000). For each change, a line is drawn between the IP prefix and the AS involved. Each line is colored according to the type of the change. On this day, there are many H-type (blue) and B-type (green) changes originating from a single AS to a few blocks of IP addresses.

correlating different events. To assist our memory of patterns from previous days, we allow a user-defined window of a certain number of days prior to the currently shown date. Data from these previous days are displayed, but with darker, less saturated colors, so that the current day's data stands out.

By choosing parameters like what IP prefixes to zoom in on, which AS numbers to focus on, which type of changes to view etc., the user can view vastly different information. Depending on the combination of chosen parameters, the user can see the overall pattern of the data, or the user can focus attention on very specific parts of the data. Different choices would reveal different anomalies and information.

3.3.3 Anomalies Detected

To detect anomalies, we are particularly looking for the following three elements because they are quite useful to visually detect anomalies.

- *Measure intensity* is defined as the total number of observed OASC instances. Normally, the number of OASC instances in the Internet is quite limited. When some faults/problems happened, the number of the colored lines would increase significantly. While it is possible that some ISPs had some dramatic network topology (or configuration) changes in one single day, it is very valuable to monitor the health of the network through this measure.

- *AS anomaly* refers to the unusual behavior per AS. It is useful to identify a small number of problematic ASes because most of the practical BGP

problems today only involve one or two ASes.

- *animation correlation* refers to the special correlation relations across the time domain.

We have identified the following anomalies by using our system and further data analysis confirms that most of them are resulted from faults.

- *Case 1:* in Figure 3.17, the entire square is covered with blue lines (H-type changes). Such an unusual scene easily caught our attention. Further data analysis confirms this is an anomaly because on that day AS 7777 falsely announced more than 4000 /32 prefixes while /32 is generally not allowed to be announced to the global Internet.

- *Case 2:* another anomaly, happening on September 18, 2001, was detected because more than 1000 H-type changes occurred that day. It is also noticed that the AS involved in those changes is using a private AS number, which confirms such occurrence is highly likely due to misconfigurations: the private AS number should never appear in the global routing system[HB96].

- *Case 3:* on Dec. 22, 2000, there are more than 10,000 B-type OASC instances observed, and all of them involve AS1221. Again, these OASC instances likely result from faults because the measure intensity is so large. The same observation was made for December 3, 2000 when AS1221 was involved in more than 1600 B-type changes, and those more-specific prefixes disappeared in the next day, which suggests an unintentional error.

- *Case 4:* the most interesting aspect of our tool is to discover "correlation" relations via the animation of the BGP data sets. Figure 3.18 shows a large number of changes due to AS 15412 erroneously announcing prefixes belonging to many different ASes on April 18, 2001. The next day, changes were made to correct the error, shown by Figure 3.19. Although Figures 3.18 and 3.19 look disorderly, an identical pattern is easily observed because the changes involved the exact same prefixes and ASes, once again demonstrating the effectiveness of human pattern recognition. In fact, this storm of on and off CSM and CMS problems has occurred since April 6, 2001. The animation helps the system administrators to discover not only that a problem has occurred but also how one type of MOAS conflict affects another type.

We have not found explanations for many of these observations. With more investigation, and further exploration with the visualization tool, we will be able to find out why these changes occurred.

3.4 Chapter Summary

This chapter presented a simple and easily deployable approach for detecting invalid MOAS cases. Instead of adding new cryptographical checks to secure routing information exchanges, our solution adds a simple MOAS list to route announcements. Whenever a prefix is announced by more than one AS, each of the ASes attaches to the prefix an identical MOAS list. If a fault results in an

invalid route announcement, the MOAS list attached to this route will be either missing or otherwise inconsistent with the MOAS list on valid announcements for the same address prefix. To prevent a router from detecting the false announcement, an attacker must block all the potential paths through which the valid route can reach the router. As demonstrated by our simulation experiments, blocking all the paths that valid routing information may take to reach a router is difficult, if not impossible, in a richly connected mesh topology such as that found in today's Internet. Our simulation results also show that the solution exhibits more robust behavior against randomly selected attackers in larger networks. One distinguished advantage of our solution is that it can be incrementally deployed in the current network using existing BGP techniques, and can effectively protect the routing system against false routes even when it is partially deployed.

However we believe that the key contribution of this work is our solution's resilience against any single point of failure. In cases where one solely relies on encryption-based techniques to secure routing information exchanges, the compromise of one router can allow the propagation of false route announcements to other routers, and such faults may not be easily detected. On the other hand with our solution, a compromised router can inject false routes into the system, but it cannot easily prevent correct routes from being propagated everywhere, thus other routers can detect the faults by noticing the conflicts between correct and false route announcements. Our solution complements encryption-based security techniques in assuring correct operation of the routing protocol in a large

scale network by adding a simple, yet robust fence against traffic hijacking by false route announcements.

The MOAS List cannot detect more complex forms of false route announcements. To move one step further, in this chapter, we also have demonstrated the principles and effectiveness of using visualization as a tool for detection of OASC anomalies. We believe that this visual-based approach will be effective in improving BGP security, although it has certain limitations.

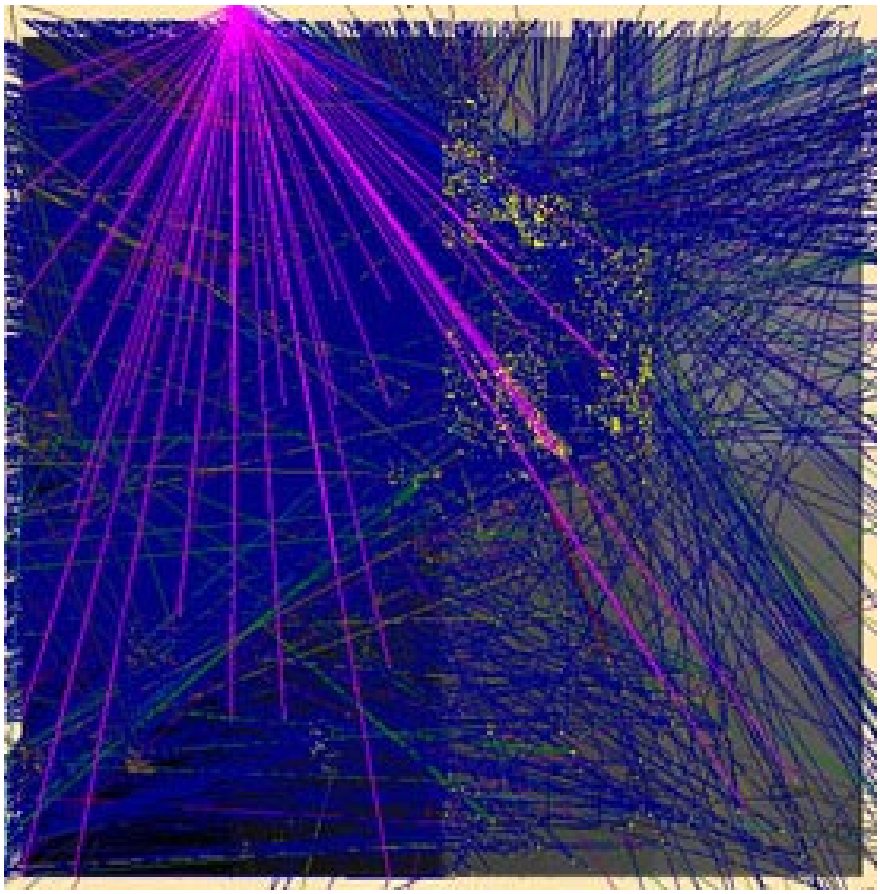


Figure 3.17: Data on August 14, 2000. An anomaly is observed despite visual clutter. Many B-type changes involving different ASes and IP prefixes occurred. Some OS-type (pink) changes are highlighted. These OS-type changes all involve AS 7777 and far-apart IP prefixes. This also indicates a fault.

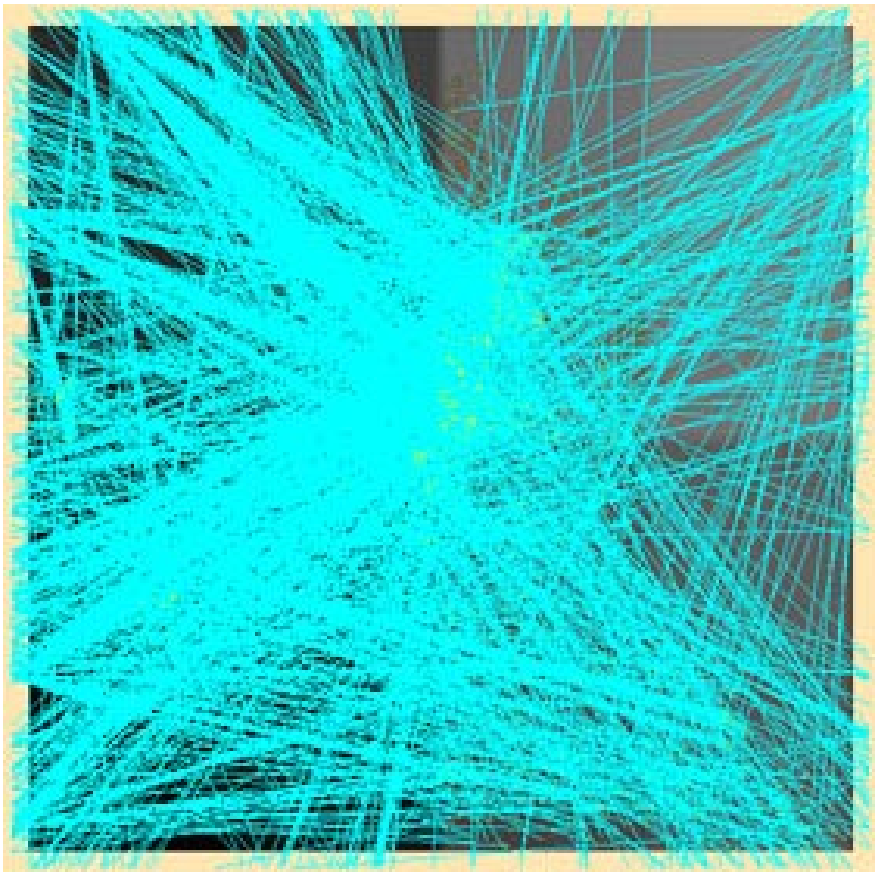


Figure 3.18: CSM-type changes on April 18, 2001.

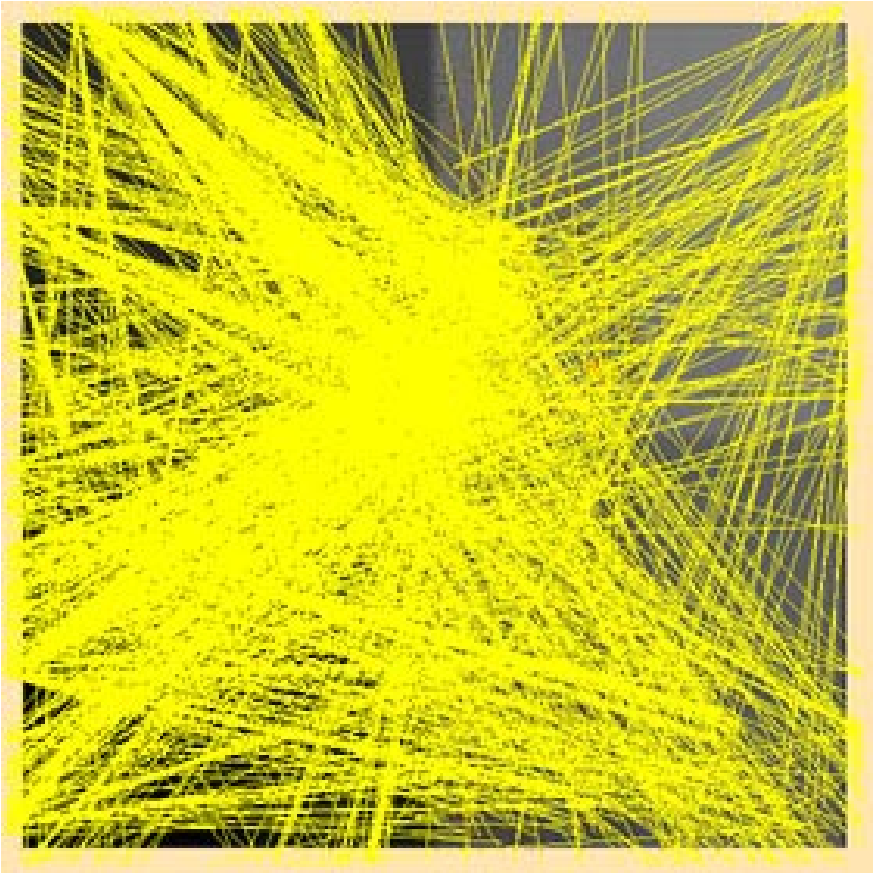


Figure 3.19: CMS-type changes on April 19, 2001. Pattern identical to CSM-type changes on the previous day (see Figure 3.18).

Chapter 4

Slow Convergence

This chapter presents a joint work with Dan Pei and other FNIISC [fni] team members. Chapter 2 briefly introduced the slow convergence problem. This chapter will review the proposed solution and describe how we use the simulation to evaluate our solution. Although the team developed the main idea, I was mostly responsible for the simuation.

4.1 Assertions for Improving BGP Convergence

The *routing convergence period* is defined as the period that starts when a previously stable route to some destination D becomes invalid and ends when the network has obtained a new stable route for D (or when D has been correctly declared unreachable). We evaluate routing convergence based on the length of the convergence period and the number of intermediate route changes that

occur during the convergence period. Due to factors such as processing and propagation delay, obtaining the new route will always require some time, and there will always be at least one route change since the previous stable (and now invalid) route must be removed. In addition, BGP specifies a parameter, commonly known as the *Minimum Route Advertisement Interval (MRAI)* or $MinRouteAdver$ timer, to determine the minimum amount of time between two successive route announcements for a same destination from a single BGP router [RL95]. The purpose of this parameter is to limit the rate of route announcements so that the routing information will not be overly announced; however, the parameter introduces longer delay for the routing convergence. Considering all these delays, we say a slow convergence problem occurs if the convergence time greatly exceeds the time that would have been required to propagate the new stable route if no invalid intermediate routes had been tried.

The key to our solution is that upon receiving a new route, we use the new route to check the existing routes so to detect and ignore invalid ones. This idea is reflected by the following two basic assertions. Dan et al. [PZW+02] has proven the correctness of these assertions.

4.1.1 Basic Assertions

Let's assume node R has neighbors $N_1, N_2, ..., N_n$. For the same destination D, R may have more than one available path. In this case, R will select one path as the *best path*, while other paths will be referred to as the *backup paths* or *backup routes*. When the best path becomes unavailable, R will re-select a new

best path among backup paths. First, we introduce some notation. Let

$$path(N_i, D) = (N_i, P_{i,1}, P_{i,2}, \ldots, P_{i,n}, D)$$

be the path from N_i to D, and

$$path_{N_i}(P_{i,j}, D) = (P_{i,j}, P_{i,j+1}, \ldots, P_{i,n}, D)$$

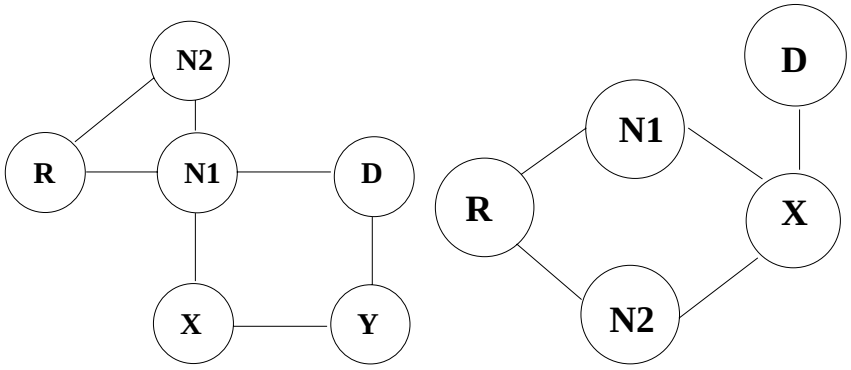
be the subpath of $path(N_i, D)$ starting from the node $P_{i,j}$.

Definition 1. Route Withdrawal Assertion. *If the latest update is that N_{lost} withdraws its path to D (i.e., $path(N_{lost}, D) = NULL$), then mark $path(N_i, D)$ as infeasible if N_{lost} appears in $path(N_i, D)$.*

That is, upon receiving a withdrawal from N_{lost}, we check whether any existing path reaches D by using N_{lost} as an intermediate node. If so, we mark it as infeasible.

Take Figure 4.1(a) as an examplel; suppose R's neighbor N_1 is advertising the path $P_1 = (N_1, D)$ and another neighbor N_2 is advertising the path $P_2 = (N_2, N_1, D)$. When N_1 withdraws its path P_1 to R, our approach says now R should mark P_2 as an infeasible path and should not select it as the best path, because P_2 is using N_1 as an intermediate node while N_1 already explicitly withdrew the part of P_2: (N_1, D).

Note that some invalid paths might be not invalidated by this assertion. Take Figure 4.1(b) as an example, and suppose the link between X and D



(a) N_1 in $path(N_2, D)$ (b) N_1 not in $path(N_2, D)$

Figure 4.1: Example network topology

fails; thus D is not reachable for X, N_1, N_2 and R. X will send withdrawals to both N_1 and N_2. After receiving X's withdrawals, N_1 and N_2 will also send out withdrawals since all of their paths go through X. Now suppose N_1's withdrawal reaches R before N_2's does. R cannot mark $path(N_2, D) = (N_2, X, D)$ as infeasible although $path(N_2, D)$ is already invalid, since N_1 does not appear in $path(N_2, D)$. Then $path(N_2, D)$ is selected as the best path by R, and if R has peers other than N_1 and N_2, $path(N_2, D)$ may be further propagated. R will recognize D as not reachable after receiving N_2's withdrawal.

Note also that the Withdrawal Assertion is not applied as the result of some link failures. For example, suppose the link between R and N_1 in Figure 4.1(a) fails. R will then remove $path(N_1, D)$, but this is not equivalent to a withdrawal from N_1. After this link failure, $(R, N2, N1, D)$ is a valid backup route that still indirectly goes through N_1.

Definition 2. Route Change Assertion. *If the latest update is that* N_{change}

advertises $path(N_{change}, D)$*, do the following:*

- *If* N_{change} *appears in* $path(N_i, D)$ *and* $path_{N_i}(N_{change}, D) \neq path(N_{change}, D)$*, then mark* $path(N_i, D)$ *as infeasible.*
- *If* N_i *appears in the* $path(N_{change}, D)$ *and* $path_{N_{change}}(N_i, D) \neq path(N_i, D)$*, then mark* $path(N_{change}, D)$ *as infeasible.*

Upon receiving a new path, one uses the new path to check the feasibility of existing paths. Conversely, the existing paths are also used to check the feasibility of the new route. Again, we use Figure 4.1(a) as an example. When the link between N_1 and D just goes down, N_1 switches to and advertises another path $P_3 = (N_1, X, Y, D)$ to R, which replaces its old path $P_1 = (N_1, D)$ and triggers R to re-select the best path to D. In this case, our approach says now R should mark P_2 as an infeasible path and should not select it as the best path, because P_2 is using N_1 as an intermediate node while N_1 already implicitly withdrew the part of P_2: (N_1, D).

4.1.2 Practical Considerations

Route Withdrawal/Change Assertions could provide mechanisms for improving the convergence of BGP. However, there are some practical concerns, such as *policy withdrawal, traffic engineering, and AS partition.* The following is the brief description of these issues.

Failure Withdrawals and Policy Withdrawals

When a neighbor withdrew a path, it is not necessarily true that the neighbor cannot reach the destination. In BGP, there are two distinct causes for a BGP withdrawal message. A *failure withdrawal* occurs if an AS has lost its route to the destination. Failure withdrawals can occur due to the failure of a route imported from IGP, the close of the peering session with the upstream peer advertising the route, or a withdrawal received from the upstream peer. In all of these of cases, the existing route to the destination is no longer valid, and the failure withdrawal conveys topology information that can be used to *invalidate* other routes(mark them as infeasible).

A *policy withdrawal* occurs if a change in route policy causes an AS to stop advertising a route to some of its neighbors. In this case, the upstream router still has its existing route to the destination, but the upstream router no longer makes this route available to some peer(s).

To determine whether a backup route is feasible, one must distinguish between failure withdrawals and policy withdrawals, because only failure withdrawals reflect the topological changes. Our *Route Withdrawal Assertion* does not apply to policy withdrawals. For example, in Figure 4.1(a), when N_1 withdraws the path $P_1 = (N_1, D)$ to R due to policy changes, R cannot invalidate the path $P_2 = (N_2, N_1, D)$ because N_1 does not withdraw the path (N_1, D) to N_2 and P_2 is still a valid path.

However, the BGP specification does not differentiate a failure withdrawal from a policy withdrawal, so we extend the BGP protocol to signal failure/policy

withdrawals. To do this, a simple 1-bit withdrawal type flag would have been enough, but there are no reserved bits left in the BGP UPDATE message. Instead, a router signals a failure withdrawal by including a *failure withdrawal community attribute(FWCA)* (0x88888888, for example)in the BGP UPDATE message. If the *FWCA* is not present, then any withdrawn routes are assumed to be policy withdrawals.

Logical AS: Signal Traffic Engineering

One AS, through multiple BGP routers, may advertise multiple routes to one single destination in the Internet. For example the Oregon Route Views Server [rou] shows that on 6/8/2001, AS 701 announced two different routes to prefix 169.131.0.0/16, in an attempt to better engineer the traffic to this destination. As a result, AS1 had learned the route (701 6079 4527) and AS1740 had learned the route (701 6347 4527). Further analysis of the data shows that 56,081 out of 121,602 prefixes, and 125 out of 11,514 ASes in the Internet are involved in traffic engineering from 07/10/2001 to 07/18/2001, which may imply the traffic engineering is a common practice today.

In general, within one AS X there are multiple routes available to a destination d. When there is traffic engineering for d in AS X, different BGP routers may select and advertise different best routes. In this case, the Route Withdrawal Assertion may incorrectly mark the valid route. For example, as shown in Figure 4.1(a), N_1 may advertise the route $R_1 = (N_1, X, Y, D)$ to R and advertise the route $R_2 = (N_1, D)$ to N_2. If this is the case, when N_1 withdraws

R_1, it is not necessarily true that R_2 is also infeasible, because N_1 is actually using two different path to reach D. Thus R should not mark R_2 as infeasible.

In this case, AS N_1 is *virtually* divided into multiple ***logical ASes*** according to the $EntryRouter$ (RID) of the selected route. In general, each logical AS in AS X is uniquely identified in the Internet by the 2-tuple (X, RID). All the BGP routers in AS X that select the best route from the Entry Router RID belong to the logical AS (X, RID). By taking a logical AS as a single AS, we still are able to apply our assertions.

Addressing the AS Partitions

In some scenarios, an AS may become partitioned into several parts due to failure of internal links. As a result, routers in different partitions could choose different routes to one destination, or some routers could lose the route to the destination while others still have the route. Internet AS partitions should be rare and be fixed quickly, but in order to guarantee the correctness of our assertions, one should not apply the assertions to any withdrawals or new route changes that resulted from AS Partition.

We assume there is already a mechanism to detect an AS partition. When an AS partition occurred, if a router lost all routes to some particular destinations and detected the AS partition, it should send policy withdrawals to its peers, since other routers in the same AS may still have the route. If the router still has a route to the destinations, it should send the new best routes to its peers by attaching its own router ID, since the AS was actually partitioned into several

logical ASes, like the traffic engineering case we mentioned above.

4.2 Simulation Results

To test the BGP convergence assertions, simulations were conducted on relatively large network topologies. The results show a substantial reduction in both convergence time and number of update messages exchanged.

4.2.1 Simulation Setup

We implemented our assertions in the BGP protocol of SSFnet simulator[ssf]. We configured all $MinRouteAdver$ timers with 30 seconds and set the link delays to be 0.01 second. The CPU processing time of each message was randomly generated during simulation to be between 0.01 and 0.05 second. The total delay for processing one message was its CPU processing time plus the sum of delays of all the other messages that arrived before this message.

To generate the topologies used in the simulation, we first obtained a BGP routing table from the Oregon Route Views server [rou]. Then we inferred BGP peering relationship based on the AS Path attributes in the BGP routes. For example, if a route to a prefix p has the AS Path (1239, 6453, 4621), we consider AS6453 to have two BGP peers, AS1239 and AS4621. We also mark AS6453 as a *Transit AS* since packets to and from AS4621 may traverse through it (note that AS1239 is also a transit AS). If an AS does not appear to be a transit AS in any of the routes, we consider it a *Stub AS*. Transit ASes are usually ISPs (e.g., AS1239 is Sprint), while stub ASes are networks at the edges of

topo. \ degree	1	$2-5$	$6-10$	>10	avg. degree
1	8	15	4	3	4.07
2	6	15	5	4	4.73
3	8	16	5	1	3.27
4	6	15	5	4	4.53
5	4	19	3	4	4

Figure 4.2: Topology statistics for network with 30 nodes

the Internet such as small organizations and universities. Next, we randomly selected $x\%$ of the stub ASes and constructed a topology containing these stub ASes and their peers, with the peering relationship among them completely preserved. We pruned transit ASes with too few peers to get the final topology. For the simulation, we modeled only one router in each AS and thus no traffic engineering or AS partition occurred. Figure 4.2 shows the topological statistics for the 30-AS network we generated for our simulation.

We ran simulations for network topology sizes 10, 15, 20, $\cdots$, 60. For each network size, we randomly generated 5 topologies; for each topology, we randomly chose one stub AS as the *origin AS* of the destination 3 times. Therefore, each data point in the results is the average of 15 simulation runs. We simulated route failure and route failover, and measured both the convergence time and the number of update messages. It should be noted that both the Route Withdrawal Assertion and the Route Change Assertion apply to all the simulations.

4.2.2 Route Failure

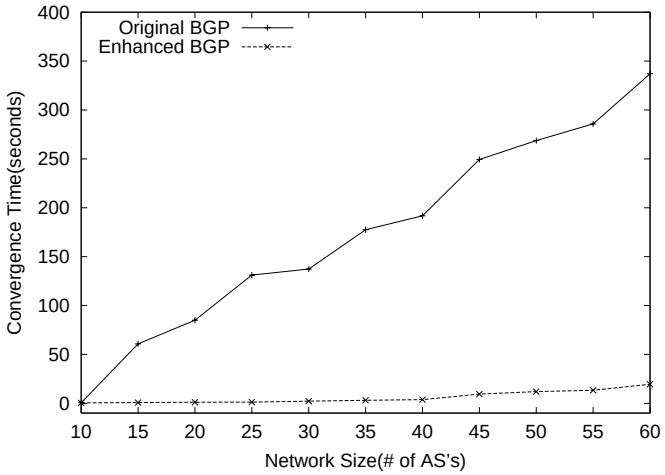
To simulate route failure, we randomly selected one stub AS with degree 1 (i.e., a stub AS with only one peer connecting it to network). This AS first announced

a route to its peer, and after the whole network had converged on this route, the stub AS withdrew the route. We compared the convergence time and number of updates of the original BGP with the enhanced BGP, as shown in Figure 4.3(a) and 4.3(b).

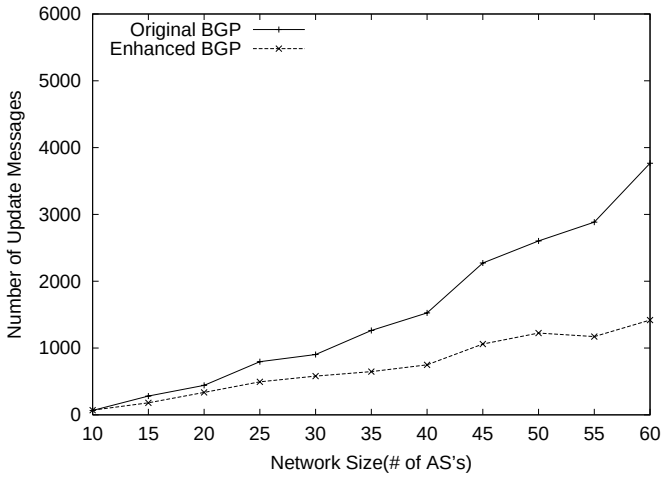
The convergence time and the number of update messages for the original BGP increases greatly as the network size increases. The original BGP explores all the backup routes before convergence. Therefore, as the network size increases, more backup routes become available and the number of route changes also grows. A new route change has to wait for the $MinRouteAdver$ timer to expire before it can be sent out; therefore more route changes leads to much longer convergence time. The convergence delay is exaggerated by the fact that more messages will incur longer CPU processing time thus longer delay.

As shown in Figure 4.3(a), the enhanced BGP improves the convergence time substantially. For example, in a 60-AS topology, the convergence time is reduced from 337.0 seconds to 19.5 seconds. However, as the network size increases, the convergence time of the enhanced BGP does increase from 0.48 second in 10-AS networks to 19.5 seconds in 60-AS networks. We attribute this to two factors. First, the network diameter also increases as the network size increases, resulting in longer propagation delay from the origin AS to the farthest AS. The second factor is that more update messages incur longer delay, and the number of messages does increase a lot for the reason described below.

Figure 4.3(b) shows that the number of messages also improves in enhanced BGP compared to original BGP. However, this improvement is not as dramatic



(a) Convergence Time



(b) Number of Update Messages

Figure 4.3: Comparison of original and enhanced BGP for route failures

as the reduction in convergence time. For example, in 60-AS networks, the number of messages is reduced from 3766 to 1419. One major reason is that removing a single update may cut convergence time by 30 seconds. A second reason is that $MinRouteAdver$ timer may have already queued and then reduced the number of updates substantially[GP01], in both original and enhanced BGP. A third reason is that even with Withdrawal Assertion, it is still possible that an invalid route is selected and propagated as shown in Section 4.1. Finally note also that our assertions only reduce the number of invalid update messages, but many valid update messages (e.g., withdrawals in route failures) still need to be propagated to achieve and confirm convergence. We confirmed the above analysis by studying simulation log files.

4.2.3 Route Failover

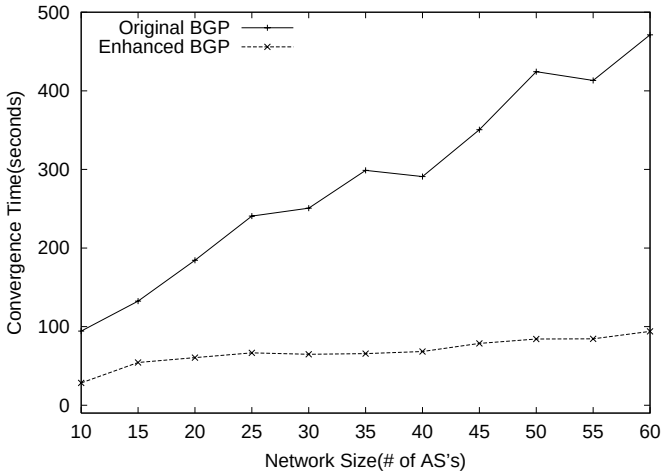
In order to measure the route change assertion, we simulated current Internet multihoming practice and created a route failover similar to the one described in [LABJ00]. We first randomly selected one stub AS with degree 2. This multihomed stub AS announced one short (primary) route to one of its two peers and announced a much longer (backup) route to the other peer. The backup route is created by prepending the stub AS's number 30 times, making the route long enough to ensure that every AS in the network would always prefer the primary route. When the primary route was withdrawn, only the backup path remained available and the routers converge to the backup path. We compared the convergence time and number of updates of the original BGP

with the enhanced BGP, as shown in Figure 4.4(a) and 4.4(b).

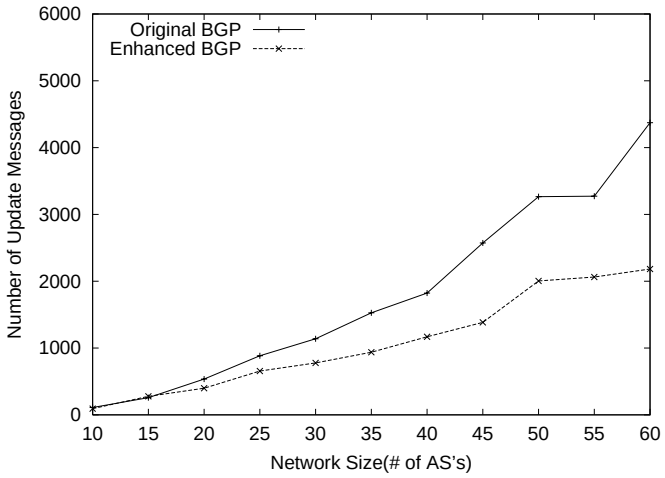
Substantial reductions of convergence time and number of messages are achieved in enhanced BGP. For example, the convergence time is reduced from 471.2 seconds to 93.9 seconds, and number of messages is reduced from 4732 to 2183 in a 60-AS network. While the convergence time for original BGP grows rapidly as the network size increases, the convergence time for the enhanced BGP grows slowly and is usually between 30 seconds and about 90 seconds. Study of the simulation log files shows that convergence time for one single simulation run is usually about a multiple of 30 seconds (note this is due to the 30 second $MinRouteAdver$ timer).

These convergence times are longer than those seen in route failure experiments. In route failover, ASes have to propagate route announcements as opposed to propagating withdrawals. This is an important distinction since route updates are limited by the $MinRouteAdver$ timer, but the withdrawals are not. Therefore, the $MinRouteAdver$ timer plays more important role in route failovers and reduces the improvement of convergence time.

This is illustrated by Figure 4.5, in which a point (x, y) represents that at time x, totally y percent of the ASes have converged. A jump of the curve means that a large number of ASes converge in a short period. The jumps for route failures in Figure 4.5(a) are more significant than those of route failovers in Figure 4.5(b), for both original BGP and enhanced BGP. In route failures, as soon as an AS converges, it can send out a withdrawal to its peers, leading to more convergence and resulting in the huge jump of the curve. However in



(a) Convergence Time



(b) Number of Update Messages

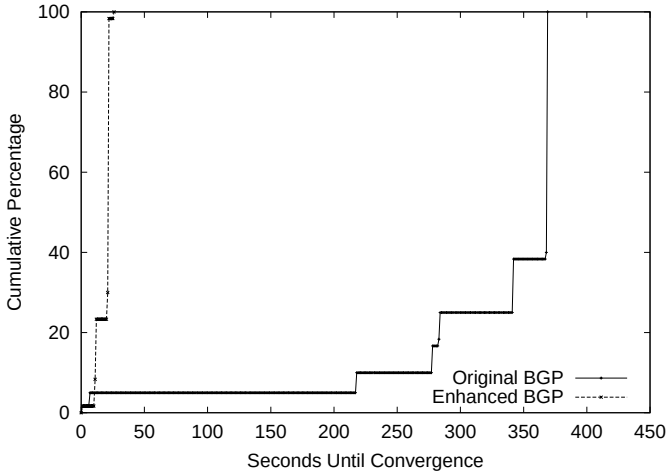
Figure 4.4: Comparison of original and enhanced BGP for route failovers

route failovers, even though an AS has converged, it cannot send the update out until $MinRouteAdver$ timer expires. This could delay the convergence of its peers by up to 30 seconds.

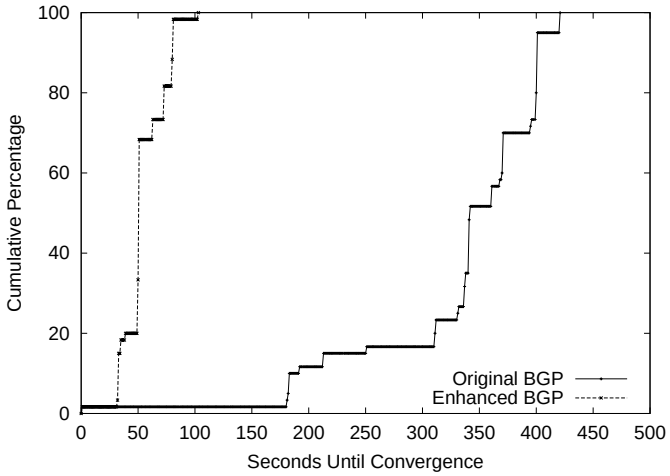
4.2.4 Discussion

There are several simplifications in our simulations. We use only one router in one AS and thus do not include the impact of IGP, traffic engineering, and AS partition. The simulated link delays may not a realistic approximate of the link delays in the Internet. During simulation, we also observed that different configurations of CPU processing delay lead to slightly different results; thus a better estimate model for it could be studied. Finally, although the network topology is derived from BGP routing table, its resemblance to the actual Internet could be studied more closely.

There is also an alternative approach, which adds the information in an AS path to explicitly specify which AS actually fails to reach the destination. Compared with our approach, which only indicates the existence of a failure in a path, this solution seems better because it provides more information. By knowing exactly which AS has the failure, a router is able to invalidate more paths which use the failed AS to reach the destination. However, such optimized solution will introduce new weakness into BGP. For instance, AS X may intentionally use other AS's number, saying for example, AS Y, as the failure point. As a result, routers using such an approach will mark all paths including Y as infeasible, which might be a Denial of Service attack to Y. The



(a) Route Failure



(b) Route Failover

Figure 4.5: Comparison of cumulative convergence percent for route failure and route failover

point of this discussion is that BGP has to be engineered to become more robust and fault resilient, which means trade-offs have to be made between different factors, such as performance and security.

4.3 Chapter Summary

Instead of blindly accepting all BGP UPDATEs, our basic approach is to let a router check route consistency using the information it has learned from previous updates and from other neighbors. In particular, in this work we used the information provided in the AS path to define route consistency assertions and used these assertions to identify infeasible routes. By taking this broader view and exploiting the relationships between routes, we were able to substantially reduce the BGP convergence time, as shown in our simulation results.

In this paper we showed how to implement our assertions in BGP and verified our design by developing a backward compatible implementation of the MRTD routing software. We demonstrated, through both simulation and measurement over our BGP testbed, that our approach reduces the BGP convergence time by up to 1 to 2 orders of magnitude.

Future work includes deriving traffic engineering information from Oregon Route Views Server and using this information in simulations and in experiments on testbeds with larger and more complex topologies.

Chapter 5

Malicious/Unstable Routing Path

This chapter presents a joint work with Lan Wang and other team members of the FNIISC project. In the first part of this chapter, I will briefly introduce the possible attacks to DNS root or generic Top Level (gTLD) servers and our countermeasures. Then I will describe the evaluation of our solution in more detail, which is my major work in this study.

5.1 Motivations and Path Filtering Approach

As described in Chapter 2, catastrophic failure of Internet services occurs when sites are unable to reach root or generic Top Level (gTLD) Domain Name servers. Catastrophic failure can also occur if a site is convinced to use an

impostor root or gTLD server. By announcing false routes, an attacker can deny access to these servers or can redirect DNS queries to a malicious impostor. Routing-based attacks have unfortunately become quite feasible [HW01].

To prevent this kind of attack, our approach is to specify the set of paths that we want to accept in the future from the peers, based on past observation of their BGP routes to the DNS servers. By accepting only these stable paths, we naturally filter out most misconfigured and transient paths. More importantly, by tracking only these paths, we can easily detect those new paths that indicate either topology changes or false routing information. We then use stronger validation mechanisms to further distinguish these two cases.

One question concerns route stabilities. If the routes to servers are too dynamic, the filtering approach may not be that efficient because the filters have to be changed frequently to adapt to the legitimate route changes, which either results in high operational costs when filters need to be configured manually, or requires our algorithm be smart enough to adapt itself to legitimate route changes well. However, we have observed in the data that routes seem quite stable. Our data show that BGP routers use primarily one or two paths to reach a top-level DNS server during a certain period of time, and that this set of paths seldom changes from one time period to another.

For example, for A root server, the path changes observed by a router in ISP 1 is shown in Table 5.1. Each path observed is assigned with a unique number, denoted by $PathNo$. In Figure 5.1, we plot $(Time, PathNo)$ pairs for the time period from 02/24/2001 to 02/24/2002, which shows the path changes

Time	AS Path	$PathNo$	Duration (seconds)
20010306 03:29:24	7018 4200 6245	1	136
20010306 03:31:40	7018 4200 6245	1	43617
20010306 15:36:21	7018 3561 6245	2	73
20010306 15:37:34	7018 4200 6245	1	184271

Table 5.1: Path Changes along the Time

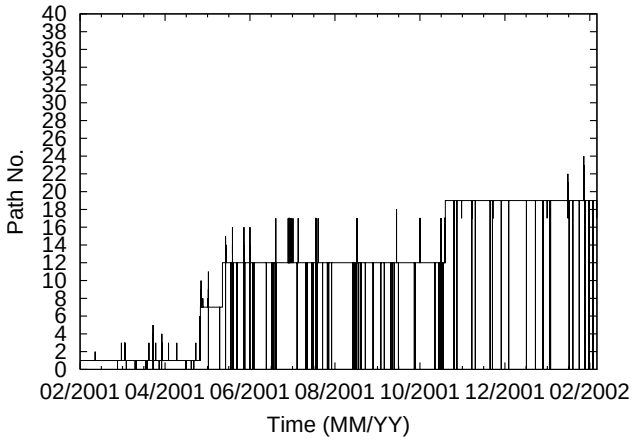


Figure 5.1: Route stability to top level DNS servers. All paths to A root server appeared in BGP data are collected and each path is assigned a unique number. In this figure, x-axis represents the time, from 02/24/2001 to 02/24/2002, and y-axis represents a numbered path. A path labeled 0 means there is no route to the server due to BGP session down or route withdrawal. In addition, only the daily largest $PathNo$ is shown in this figure due to its resolution.

along the time for a particular root server from a single router's point of view. In this figure, x-axis represents the time, from 02/24/2001 to 02/24/2002, and y-axis represents the numbered path. A path labeled 0 means there is no route to the server due to either the BGP session being down or routes withdrawn. In addition, only the daily largest $PathNo$ is shown in this figure due to its resolution.

This figure shows that only several primary paths were mainly used during

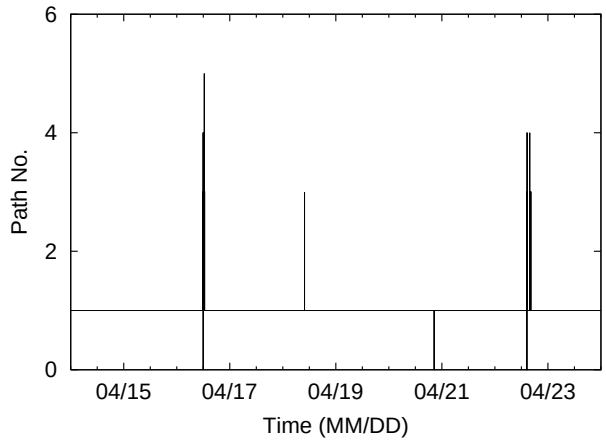


Figure 5.2: Route stability to top level DNS servers for a 10-day period.

the one year period. The other paths are either backup paths, which may indicate the primary path failed, or transient paths, which may be due to slow convergence or misconfigurations. When we take a closer look at the data for a randomly selected 10-day period from 04/14/2001 to 04/24/2001, as shown in Figure 5.2, it appears that for a short time period, only one or maybe two primary paths were mainly used.

One reason that routes to top level DNS servers are quite stable is because those servers are normally associated with high operational expectations, compared with random sites. Consequently, the servers are normally located in a place with stable and reliable Internet connectivity, which results in the stable path.

It is inevitable that the filtering approach introduces short delay in adapting to topology changes. However, due to the redundancy of the top-level DNS servers, the routing adaptation delay to one or a few servers would not have a

major impact on the overall reachability to the DNS servers. Therefore, we are confident that the security provided by our approach far outweighs the effect of its associated adaptation delay.

Some other approaches, such as SBGP [KLS00] and DNSSEC [MR01], may gradually make it difficult for a false route or TLD server to be inserted, but they require significant modifications to current BGP/DNS implementations. On the contrary, our approach is simple and readily-deployable because we utilize an existing BGP mechanism (AS-path filtering) that is in frequent use in Internet routers.

5.2 Design of Path Filters

The objectives of DNS path filtering are two-fold. First, since invalid routes to the top-level DNS servers could potentially affect millions of Internet users, we want to filter out those routes with very high probability. More specifically, there are two types of BGP paths that we want to filter out: (1) paths injected by malicious attackers or by misconfigured routers; and (2) obsolete transient paths such as those introduced by BGP slow convergence [LABJ00]. Second, we want to ensure that reachability to the DNS servers will not be jeopardized by path filtering, i.e., routers should still adopt valid new routes when topology or policy changes occur.

A path filter (F) is a set of BGP paths. For example, a path filter for DNS root server F root server may consist of two paths p_1 and p_2, i.e., $F = \{p_1, p_2\}$,

where p_1 is (2914 6461 3557) and p_2 is (7018 701 3557). When a router receives a BGP update for the address prefix of a protected DNS server, it checks if the received AS path matches any path in the path filter. If so, it will accept the update for routing decisions. Otherwise, it will ignore the BGP update (i.e., the path is filtered out)[1]. In our previous example, the path filter restricts the router to use only two paths ((2914 6461 3557) and (7018 701 3557)) to reach the f root server. Therefore, if any AS other than AS3557 is misconfigured to originate a path to 192.5.4.0/23 or an attacker forges a path to this destination, the router will ignore these invalid paths and therefore will not be affected by them.

More specifically, the path filtering algorithm will have three components. We use a *monitoring process* to identify suspicious paths and a *verification process* to further examine the validity of the suspected path. A *filter construction process* then dynamically adjusts the filters based on the BGP path statistics collected by the monitoring process and the feedback from the verification process. To ensure the routing and forwarding performance of BGP routers, these processes are intended to be light-weight background processes. In addition, the verification functionality can be performed by a separate machine which may simultaneously support multiple BGP routers to avoid redundant verification of the same BGP path, thereby reducing the overall overhead. Since the filtering mechanism already exists in today's BGP implementations, all we need to do is to construct appropriate path filters to protect the top-level DNS servers. In

[1]We assume that the update will be recorded in the incoming routing table even if it is filtered out.

the following sections, we briefly describe how our scheme works.

5.2.1 A Simple Path Filter

We adjust path filters at time intervals of length T. During each time period, we monitor the usage of every path, i.e., the percentage of time it is announced by a peer, regardless of whether it is currently in a filter. Each path's usage is recorded by $U_k(p) = T_k(p)/T$. At the end of a time period, we consider only paths that have been used more often than a configured threshold, i.e., the *base paths*, as candidates for the path filters used in the next time period. This is to screen out misconfigured and transient paths. We then verify these candidate paths to see if they are indeed valid. Since attackers may pose as legitimate DNS servers for a period of time before launching their attacks, we also verify previously trusted paths with a probability P_v, Only paths that have appeared as base paths and that have passed the verification test will be placed in the filters.

The presudo-code for such a simple path filter construction process is described as the following:

Algorithm 5.2.1.

```
1 Algorithm to Adjust a Filter
2 while in_current_time_period()
4       comment: Receive validated new paths from the verification process
5       p = recv_new_path();
6       F = F ⋃ p;
```

```
7         p.checked = 1;
9   end
11  comment: Adjust F at the end of a time period
13  F_old = F;
14  F = ∅;
15  foreach p ∈ F_old
16    if U_k(p) ≥ U_min
17      comment: Consider only base paths
18      then
19            if p.checked = 1
20              then
21                    comment: p is already validated
22                    F = F ⋃ p;
23               else
24                    comment: validate p with a probability of P_v
25                    if rand() ≥ P_v ∨ verify(p) = 1
26                      then F = F ⋃ p;
27                    fi
28            fi
29    fi
30    p.checked = 0;
31  end
```

5.2.2 An Adaptive Path Filter

When using the simple path filter, a legitimate path, however, could be delayed for a substantial amount of time before being adopted if we adjust filters only at the end of a time period. Therefore, the filters cannot adapt to the valid network topological changes promptly, which may endanger the reachability to DNS root servers.

As an enhancement to the above algorithm, we also adopt *valid* paths into the path filters if they have been advertised for a period of T_r in the current time period. T_r is typically much smaller than T and, an appropriate value of T_r should ensure quick adaptation to topology changes while minimizing the impact of transient paths. If it turns out that these paths do not meet the standards for base paths, they will be eliminated from the path filters at the end of the time period. Note that such paths also need to be validated by the verification process.

For the simple path filter, if a path is oscillating between extreme values of $U_k(p)$ and the filter selection is based solely on this measure, the path may frequently get in and out of the filter at the wrong times (as we predict the new filter based only on the most recent statistics). Therefore we also use $U(p)$, an exponentially weighted moving average (EWMA) of $U_k(p)$, to select base paths. $U(p)$ is calculated using the following formula

$$U(p) = \alpha * U(p) + (1 - \alpha) * U_k(p)$$

where $\alpha \in (0, 1)$. We consider a path as a base path only if $U(p)$ or $U_k(p)$ is greater than the minimum threshold U_{min}. Since T_r is strictly smaller than $U_{min} * T$, all the base paths must have been placed in the filter by now, but not all the paths in the filter are base paths. Therefore, we only need to select the base paths from the current filter and they become candidates for the filter used in the next time period.

5.3 Evaluation

In this section, we evaluate our adaptive filtering scheme against real BGP data. We mainly focus on its impact on top level DNS servers reachability and its effectiveness in filtering out malicious/unstable routes.

5.3.1 Parameter Setting

In our simulation, we have three major parameters which could be tuned in one way or the other depending on the desired level of security and the projected overhead.

1. T specifies the length of a time period, which may be on the order of days, weeks or even longer. In general, when a smaller T is chosen, more frequently the verification process will be invoked, which implies a larger overhead, but stronger security will be obtained. Moreover, to prevent attackers from being able to predict when a router will perform the filters construction, one should introduce some randomness when setting T. T

is set to one week in our experiments.

2. U_{min} specifies the threshold for a path can be considered as a base path. When a larger value for U_{min} is chosen, it requires a path being used by a peer for longer time; thus stronger security could be obtained because more temporary or short-lived attacking paths will be filtered out, but as a penalty, we may also filter out some valid backup paths which adversely affects the reachability to DNS servers. We choose a U_{min} of 10% in our study.

3. T_r specifies the holding time for a filtered path. After T_r, it will be treated as a regular path which might be selected for packet forwarding. A small T_r will cause a path to be quickly adopted (when $T_r = 0$, filters will be no any use); if the path is a valid one, smaller T_r will improve the reachability. However, in the meantime, it opens the door for an attacking route to be adopted quickly also. In our simulation, T_r is set to 1 hour.

4. Using an EWMA measure $U(p)$ of a path allows an already used path to stay in the path filters longer even though it may not be used for a particular period. The reason to use EWMA measure is because an old path has been working for a while thus is less likely to be a bad route. Suppose path p is now valid anymore and its current $U(p)$ is 1, then this path will be kept in the filter set for a period of $\lceil \log(U_{min}) / \log(\alpha) \rceil \times T$ before eliminating it from the filter. When other parameters are fixed, a larger α will cause an old path to stay in the filter longer, which may

T	T_r	U_{min}	α	P_v
1 week	1 hour	10%	0.75	0.1

Table 5.2: Parameter Setting

be beneficial to the servers reachability, but in turn, it is possible that, before we detect the path is no longer valid, a malicious attacker injects this already invalid path into the network. In our simulation, $\alpha = 0.75$.

Table 5.2 shows the parameters we used in our simulation. The values for each parameter are selected based on both the discussions with network operators and our experiences.

5.3.2 Data Source and Methodology

To overview the method we used to evaluate our filtering algorithm, first we collected those BGP routing updates which were used to reflect routing changes to top level DNS servers. From those updates, we are able to reveal the history of how those DNS servers were reached in the past and how good or bad such reachability was. Then we will apply our filtering algorithm to these historical BGP updates to see how our approach will affect the servers reachability. Furthermore, we already know some BGP faults, such as invalid MOAS of a DNS root server occurred at a particular time, so we can test how our algorithm will respond to such faults.

In detail, we collected raw BGP routing updates from RIPE NCC[ris] from Feb. 2001 to Feb. 2002. There are eight data collecting points (rrc00 - rrc07)

at RIPE NCC. Among them, we intentionally chose rrc00 because it receives full routing tables from its peers. Therefore, when rrc00 observed that one of its peers doesn't have a BGP route to a particular DNS server, we know that it's because this peer doesn't have a route, as opposed to because the local policy at this peer prohibits it from announcing this route. For the same reason, we selected nine peers of rrc00 that export their full routing tables during the entire examined period. As Table 5.3 shows, these routers are located not only in different ASes (some are located in tier-1 ISPs and others are in regional ISPs), but also in different countries (US, Japan and three European countries). Therefore, we were able to evaluate the filtering scheme against networks with a wide range of characteristics.

Location	ASes that rrc0's peers belong to
US	AS7018 (AT&T), AS2914 (Verio)
Netherlands	AS3333 (RIPE NCC), AS1103 (SURFnet), AS3257 (Tiscali Global)
Switzerland	AS513 (CERN), AS9177 (Nextra)
Britain	AS3549 (Global Crossing)
Japan	AS4777 (NSPIXP2)

Table 5.3: RRC0's peering ASes that we examined

There are 13 DNS root servers and 13 gTLD servers, denoted by English letters, such as A root server and B gTLD server. We first identified the longest address prefix that matches the IP address of each server (note that as a result, A and J root servers have the same BGP-visible address prefix 198.41.0.0/24). Then we extracted the BGP updates related to these prefixes from the raw BGP updates we collected . Special considerations have been taken for the gTLD

servers whose IP addresses changed during the examined period. For example, the A gTLD server changed its IP address from 198.41.3.38 to 192.5.6.30 on March 26, 2001. Through discussions with operators of Verisign, we learned that the gTLD servers located at the old addresses would be kept in operation for up to 3 days to ensure DNS reliability. Therefore, for each address change, we used a 3-day overlapping period during which BGP updates for both the old and the new prefix would be included in our data set.

We noticed that the BGP sessions between rrc00 and its peers were reset more frequently than expected, which is suspected to result in some measurement artifacts. Unlike a regular eBGP session that is established between routers sitting on the same physical subnet, these sessions cross multiple network hops (multi-hop eBGP sessions) and thus they are more susceptible to network congestion, but it is impossible for us to determine exactly what caused the session restarts from the data available. Neither can we determine what a peer's routes looked like during the session down time. The peer might have lost the route to DNS servers due to network congestion, or it might still be able to reach the DNS servers. Therefore, we removed session down periods from our calculation. We may have introduced some bias by doing so (e.g., over-estimate the reachability), but the data show that session down periods are too short to have any significant effect on the overall results.

In the evaluation, we actually simulate a router in one single-homed AS that peered with only one of the ASes in Table 5.3 during the examined period, as shown in Figure5.3. Therefore, reachability here refers to the simulated router's

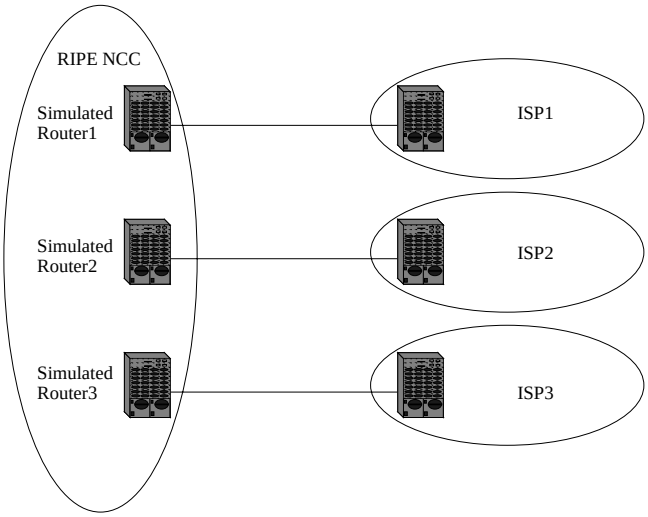


Figure 5.3: Simulated Routers

reachability to the DNS servers through the particular peer.

For each simulated router, we apply our filtering scheme to the BGP updates received from the corresponding peer, then calculate the new reachability. In detail, for simulated router R_i, it receives BGP updates from its peer P_i, and each update will contain a timestamp which indicates when the update is received by R_i. All of the updates R_i will be ordered along the timeline according to their timestamps, the earliest one first. The timeline will be divided equally at T intervals, as shown in Figure 5.4. Each box in Figure 5.4 represents the duration of a route. For example, the route $p1$ lasts for $t1$ time. Each route will be compared with the paths in the path filter, and those paths not appearing in the path filter will be marked as filtered. Let us assume we already have a path filter including $\{p1, p3\}$, the path $p2$, $p4$, $p5$ will be marked as filtered; consequently, the time $t2$, $t4$, $t5$ will not be counted for servers reachability.

More precisely, during the first time period, the simulated router only can reach top level DNS servers at the percentage of $(t1+t3)/T$. Comparably, the servers reachability without filtering will be $(t1+t2+t3+t4)/T$. We would like to point out that we derived reachability from BGP routing updates, not from DNS replies or ping-like packets. Although route availability in general leads to reachability, it is not true when the announced paths are in fact obsolete (e.g., paths announced during slow convergence).

Note that the route $p5$ crosses two different time periods, we just regard this case as two identical paths appearing in each time period. At the second period, the router $p5$ will not be counted for the reachability until it lasts longer than T_r. At the end of the second period, the path filter is reconstructed to include mostly used path $p5$ and $p6$ during this period. If $p1$ and $p2$ are different from $p5$ and $p6$, although they don't appear at the second period, they still stay in the filter path at the third period because their EWMA value may still be large enough.

In the simulation, we don't have a good way to simulate the verfication, because we cannot go back in time to verify the validity of a path, for example, using the proposed verification method. Therefore, we have to accept a new path if it persists. In other words, the monitoring process still identifies those new paths that exist long enough to warrant further checking, but the verification process will accept all of them as valid in our simulation. We can, however, evaluate how effective the filter acts as a first barrier against spoofed paths caused by router misconfiguration as well as obsolete paths due to slow routing

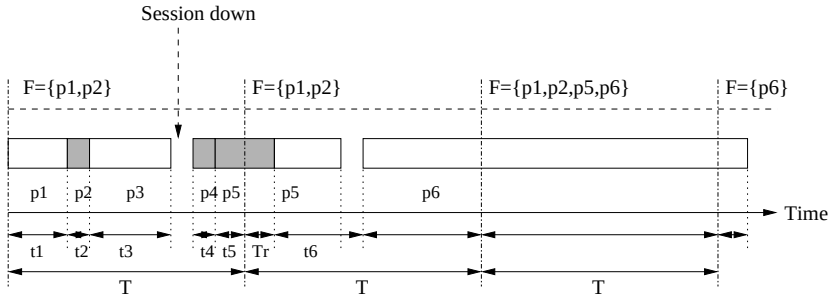


Figure 5.4: Calculation of Servers Reachability. In this figure, each box represents the duration of a route. For example, the route $p1$ lasts for $t1$ time. Those paths not matching in the path filter will be marked as filtered, shown as a grey box. For the first period, when $F = \{p1, p3\}$, the simulated router only can reach top level DNS servers at the percentage of $(t1 + t3)/T$. Comparably, the servers reachability without filtering will be $(t1 + t2 + t3 + t4)/T$.

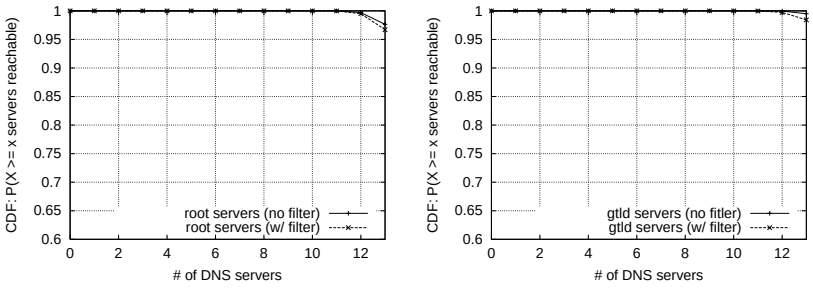
convergence.

5.3.3 Impacts on Server Reachability

In this section, the difference will be compared between the reachability with and without our filtering scheme, in order to see how the filter affects the reachability of the simulated router. We first show the results for two peers – ISP1 and ISP2; then we summarize results for the other peers. Also, the adaptive path filters algorithm is used for the following results.

Figure 5.5(a) shows the simulated router's reachability to the DNS root servers when it peers with ISP1. The basic observations are:

- During the one-year period, the simulated router has 100% of the time to reach at least 8 root servers with and without filtering. For 99.997% of the time, the simulated router can reach at least 10 root servers with and without filtering.



(a) ISP1 (root servers)

(b) ISP1 (gTLD servers)

(c) ISP2 (root servers)

(d) ISP2 (gTLD servers)

Figure 5.5: Adaptive Path Filter's Impacts on Servers Reachability. The x-axis is the number of servers and y-axis is percentage of time x or more top-level DNS servers are reachable. The solid and the dashed curves correspond to the reachability before and after we apply the filter, respectively.

- For x equal to 11, there is a very small difference: 99.982% with filtering and 99.997% without filtering. In other words, the amount of time during which at least 11 root servers are reachable decreases by 1.3 hours (accumulated over the entire year) after we apply the filter. The difference becomes 0.948% (83 hours) for the total 13 root servers.

Given that the total period is one year, the effect of filters can be considered negligible. At least, the simulated router maintains reachability to at least 8

root servers all the time, which is the strong evidence to show that the filters will not jeopardize the servers reachability. According to Figure 5.5(b), the same conclusion also applies to gTLD servers.

ISP2's overall reachability to the root servers was shown in Figure 5.5(c)), which seems not as good as ISP1's. One reason might be because for several times the routes to *all* the DNS servers were explicitly withdrawn by this peer at about the same time and then announced shortly after (note that there was no session restart during meanwhile). Totally there was 503 seconds during which no routes to any of the DNS servers were available. It is not clear to us yet why the router exhibited such behavior.

The difference between reaching at least one root server with filtering and without filtering is only 2.1 hours. To reach at least 12 root servers, the difference becomes 60 hours and 97 hours to reach all of 13 root servers. Again, according to these results, we can conclude that the filter does not have a major effect on ISP2's reachability.

No. Servers	N=1		N=6		N=13	
	Unfiltered	Filtered	Unfiltered	Filtered	Unfiltered	Filtered
ISP1	100%	100%	100%	100%	97.649%	96.701%
ISP2	99.998%	99.974%	99.972%	99.939%	99.398%	97.039%
ISP3	100%	100%	100%	100%	95.149%	93.974%
ISP4	100%	100%	99.940%	99.855%	98.017%	95.981%
ISP5	100%	99.978%	99.951%	99.947%	91.341%	90.228%
ISP6	100%	100%	99.397%	99.397%	98.808%	97.248%
ISP7	100%	100%	99.998%	99.966%	99.535%	97.395%
ISP8	100%	99.996%	99.975%	99.971%	25.728%	25.419%
ISP9	100%	100%	99.994%	99.992%	99.475%	97.780%

Table 5.4: Percentage of Time when N or More DNS Root Servers are Reachable

The results for all the nine peers are summarized in Table 5.4. We only present the reachability to the root servers here since the filter's impact on the reachability to the gTLD servers is similar and are thus omitted for brevity. One can make the following observations:

1. With filters being applied, there are 6 out of 9 peers allowed the simulated router to reach at least one root server 100% of the time, which means that from Feb. 2001 to Feb. 2002, if our filters were applied to the routers which peered with these 6 ISPs, these routers would have no problem in reaching at least one DNS root servers at any time.

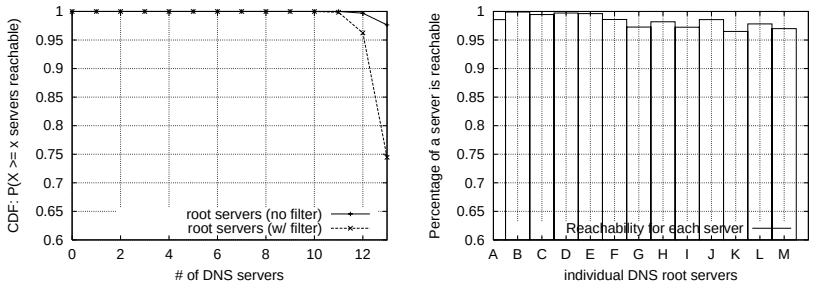
2. If a router peered with any of the other three ISPs, it is possible for the router to be not able to reach any DNS root servers for some time. However, comparing with the total one year time period, the accumulated unreachability time caused by filters is quite small: 2.1 hours for ISP2, 1.89 hours for ISP5 and 21 minutes for ISP8.

3. ISP1 and ISP3 allowed the simulated router to reach at least 6 root servers 100% of the time even after we applied the filter, while the same statistic for the other ASes' ranged from 99.855% to 99.992%.

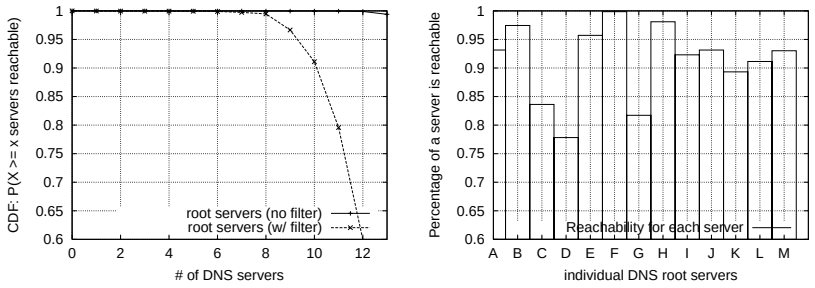
One may notice that ISP8 has very poor reachability to reach all 13 root servers. Further examination of the data shows that ISP8 cannot reach E root server for 71.5% of the time.

The evaluation against real BGP data shows our adaptive filter algorithm has very little effect on the reachability to top level DNS servers. Figure 5.6

also shows the effects caused by the simple filter algorithm.



(a) ISP1 (root servers) (b) ISP1 (individual servers)



(c) ISP2 (root servers) (d) ISP2 (individual servers)

Figure 5.6: Simple Path Filter's Impacts on Servers Reachability. In the right side of graphs, the x-axis is the individual root server and y-axis is percentage of time the server x is reachable.

Figure 5.6 shows that the filters have great effects when one tries to reach more than 6 root servers. It is because the simple path filter algorithm cannot adapt to the network routing changes quickly; thus many valid backup routes might be filtered out. However, due to the redundancy of DNS root servers, one still is able to reach at least one root server almost all of the time. The similar results are observed for other ISPs as well.

5.3.4 Effectiveness

To evaluate the effectiveness of our filter algorithm, i.e., how effective our algorithm could filter out attacking routes or transient routes resulting from slow convergence, we first show how our algorithm responds to a known routing fault. Then we identify a slow convergence occurrence in our data set and examine how our algorithm handle it.

MOAS

On April 6, 2001, a small ISP mistakenly originated thousands of prefixes including the prefix where C gTLD server is located. Those announcements were propagated to the Internet and were selected as the best path by many ASes (including some tier-1 ISPs). Out of the 9 peers we observed, 4 of them selected the wrong path as the best path. 3 peers used the wrong path for no longer than 20 minutes, and 1 peer used the wrong path for more than 3 hours.

All the short-lived announcements were filtered out by the simulated router while we accepted the one that remained longer than T_r in the absence of a verification process (T_r is set to one hour in this case). One could imagine that, without the filter, DNS requests to the C gTLD server would be forwarded to the wrong place by the simulated router and could possibly overwhelm the small ISP that injected the wrong route. With the filter, however, these requests would be silently dropped since the simulated router would have no route to the server as the single available route is invalid. The important thing is that the false route would not be further propagated to "poison" other routing tables.

In a more realistic situation where we have multiple peers, there may be valid alternative routes to choose from, and the false route would not cause the server to be unreachable. As for the one wrong path that persisted, it could have been easily identified as false if the verification process was implemented. With the help of the filter, even a network administrator can detect this error.

BGP Slow Convergence

As introduced in Chapter 2, BGP slow convergence [LABJ00, LWVA01] occurs when the current path is withdrawn and BGP explores every feasible path until it finds all of the path invalid. An efficiency enhancement has been introduced in Chapter 4.

Time	BGP Path
06/19/2001 12:35:30	3549 19836 19836 19836 19836
06/19/2001 16:06:32	3549 10913 10913 10913 10913 10913 10913 10913 19836
06/19/2001 16:06:59	3549 1239 10913 19836
06/19/2001 16:07:30	3549 701 10913 10913 19836
06/19/2001 16:08:30	Withdraw
06/19/2001 16:15:55	3549 19836 19836 19836 19836

Table 5.5: Slow Convergence (AS3549)

Table 5.5 shows the BGP update sequence in a typical slow convergence scenario: AS3549 continued in vain to explore all the available paths to A root server, until finally it gave up searching for a path and sent out a withdrawal. One may argue that such a sequence might also occur when the link between AS3549 and AS19836 was down, but based on the evidence that the other eight peers also experienced path changes for the same prefix at the same time and

most of them did not use AS3549's path to reach the server, we concluded that it was because AS19836 could not reach this prefix and thus explicitly withdrew it from all its peers.

These short-lived announcements are undesirable because they could lead to a cascade of BGP updates sent out by AS3549's peers and beyond and slow down routing convergence. We found that this slow convergence incidence produced 24 unnecessary BGP updates from the nine peers, and the simulated router was able to filter out all of them. In other words, if deployed in the current Internet, this filter can effectively prevent the propagation of these updates, thus reducing the processing load of BGP routers and improving the BGP convergence time.

5.4 Chapter Summary

DNS service is an critical component of the Internet that is used by all end users and by almost all types of applications. Catastrophic failure can occur if a single site is convinced to use an impostor root or gTLD server. As of today, such failures may easily happen due to route hijacking caused by either faults or attacks [HW01, ZPW+01]. Unfortunately, so far there has been no defined way to tell which route announcements are valid and which are not.

In this work we proposed a route filtering approach to make DNS service more resilient to faults or attacks. We have validated our design against the analysis of one year of BGP route logs. Our results show that routers using our simple AS path filtering can effectively detect the insertion of invalid routes,

while maintaining reachability to the top level DNS servers.

Our solution exploits the special properties found in the design and deployment of DNS top level servers, such as redundancy and high operational expectation. Redundancy ensures the availability of DNS top level servers even when some valid routes are filtered out, and the high operational expectation results in the relatively stable routing path to those servers. We believe our solution could be useful to those systems which have similar properties. Or one can utilize the similar filtering approach for more applications when trade-offs have been made between security and service availability.

Chapter 6

Conclusions and Future Work

The objective of this dissertation is to study the issues related to improving overall Internet security, especially from the global routing infrastructure perspective. As a critical component of the Internet, the inter-domain routing system is vulnerable because of the vulnerabilities inherited from the protocol design and potential threats from outsiders and insiders. The insider attacks or mis-configurations are especially difficult to be detected and prevented because of the limitations of the underlying algorithm BGP is using, the scalability issues of the Internet, and dynamic behaviors of BGP.

Previous secure routing related studies relied mostly on encryption-based techniques. However, this kind of approach often requires significant modi-

fications to the BGP protocol and implementations. This greatly limits the practical use of such approach when one considers that BGP is deployed widely in the Internet. In addition, the compromised router still can propagate the false route announcements to other routers, and such faults may not be easily detected.

6.1 Dissertation Contribution Summary

In this dissertation, I work on the solutions which can be integrated with the existing BGP system. First, BGP vulnerabilities are analyzed from different perspectives. Several cases have been identified which may endanger the BGP routing system and overall Internet security. We provide solutions under the same framework and operations as BGP. Equally important, the effectiveness and the associated effects of the proposed solutions are fully evaluated via experiments and simulations.

In Chapter 3, the MOAS problem is studied. MOAS could be problematic because, on the one hand, MOAS can occur for valid reasons such as multi-homing, and on the other hand, router mis-configurations have also produced MOAS. Large scale network outages and other problems have been associated with MOAS problem. To determine whether MOAS is the result of a fault or a valid operation, a simple enhancement to BGP is proposed, which enables BGP to detect invalid MOAS. This approach utilizes the rich interconnectivity of the current Internet, which makes the solution resilient against any single point of

failure, because a compromised router can inject false routes into the system, but it cannot easily prevent correct routes from being propagated everywhere. Thus other routers can detect the faults by noticing the conflicts between correct and false route announcements.

To further validate more complex forms of invalid route announcements from the faulty origin ASes, the Origin AS Change (OASC) problem was introduced and studied, which attempts to identify the abnormal OASC cases. A prototype of visual-based OASC anomaly detection system is developed to identify anomalies.

In Chapter 4, BGP slow convergence problem is introduced. To speed up the convergence, an assertion-based approach is proposed and evaluated via simulations. The results showed that BGP convergence had been greatly improved by adopting our approach.

Chapter 5 demonstrates how to improve overall Internet security by protecting top-level DNS servers from route spoofing attacks. Such attacks could cause catastrophic failure of the Internet. The path filtering approach is proposed, and its effects on server reachability is fully evaluated by analyzing one year of historical BGP data. The results showed that our approach is effective, and the effects are negligible.

6.2 Future Work

The work presented in this dissertation is one of the first steps toward a more fault resilient Internet. We do not assume that the approaches described in this document alone can provide a complete solution. Rather, a truly fault resilient system must build multiple fences against potential unknown attacks since no single fence can be strong enough to defend all. Following this direction, my future plans include:

- to identify more BGP-related weakness from different perspectives, such as policy conflicting, protocol interactions, and router attacks.

- to study how to utilize strong security protections, like proposed in SBGP [KLS00], in a practical way. One possibility is to use fault detection techniques to detect faults or anomalies first, then invoke SBGP-like services to validate received information. Therefore, we could possibly avoid the validation of every incoming route updates, which is considered as a big overhead for the routing system, and in the meantime, the stronger security and robustness may be obtained.

- to study the issues and applications of how to utilize Internet characteristics to improve BGP and the Internet security. Based on the work with MOAS problem, which exploits the rich interconnectivity of the Internet, I feel we may be able to further enhance BGP robustness and Internet security by using power law[FFF99] [CCGJ02] and other characteristics found in recent Internet studies.

- to study the issues about how the information visualization techniques, or other techniques could help us to solve BGP problems. Due to large data volume, diverse and hidden local policies, and measurement limitations, even understanding BGP behavior is quite a challenge. Can we combine human intelligence and computer computation power to solve some complexity encountered in BGP data analysis?

- to study the fundamental issues about the approach of "patching" BGP. In this dissertation, we identified two commonly used techniques: information consistency checking and filtering. The future research questions could be: can we further apply these two techniques to other BGP weaknesses? Are there any other useful techniques that could be possibly exploited?

Bibliography

[AA01] D. Atkins and R. Austein. Threat Analysis Of The Domain Name System. Technical report, November 2001. Internet Draft draft-ietf-dnsext-dns-threats-00.txt.

[B+97] R. Barrett et al. Routing Snafu Causes Internet Outage. *ZDNet*, 1997.

[BBLR98] T. Bates, R. Bush, T. Li, and Y. Rekhter. DNS-based NLRI origin AS verification in BGP. Internet Draft, Work in Progress, IETF, 1998.

[BGJ+95] T. Bates, E. Gerich, L. Joncheray, J-M. Jouanigot, D. Karrenberg, M. Terpstra, and J. Yu. Representation of IP Routing Policies in a Routing Registry. RFC 1786, March 1995.

[Boe00] L. Boettger. The Morris Worm: How it Affected Computer Security and Lessons Learnd by it, December 2000. `http://rr.sans.org/malicious/morris.php`.

[cai] The CAIRN Testbed. `http://www.cairn.net`.

[CCGJ02] Qian Chen, Hyunseok Chang, Ramesh Govindan, and Sugih Jamin. The Origin of Power Laws in Internet Topologies Revisited. In *INFOCOMM*, 2002.

[Che00] E. Chen. Route Refresh Capability for BGP-4. RFC 2918, SRI Network Information Center, September 2000.

[cis01] Cisco Security Advisory: Cisco IOS BGP Attribute Corruption Vulnerability, May 2001. `http://cio.cisco.com/warp/public/707/ios-bgp-attr-corruption-pub.shtml`.

[COPY01] J. Cowie, A. Ogielski, B. Premore, and Y. Yuan. Global Routing Instabilities during Code Red II and Nimda Worm Propagation. Technical report, September 2001. `http://www.renesys.com/projects/bgp_instability`.

[CRKGLA89] C. Cheng, R. Riley, S. Kumar, and J.J. Garcia-Lunes-Aceves. A Loop-Free Extended Bellman-Ford Routing Protocol Without Bouncing Effect. In *Proceedings of ACM SIGCOMM*, pages 224–236, August 1989.

[CS00] R. Chandra and J. Scudder. Capabilities Advertisement with BGP-4. RFC 2842, SRI Network Information Center, May 2000.

[CTL96a] R. Chandra, P. Traina, and T. Li. BGP Communities Attribute. RFC 1997, August 1996.

[CTL96b] R. Chandra, P. Traina, and T. Li. BGP Communities Attribute. RFC 1997, SRI Network Information Center, August 1996.

[Eas99] Donald Eastlake. Domain Name System Security Extensions. RFC 2535, IETF, March 1999.

[ep] Public Exchange Points. `http://www.ep.net`.

[ERH92] Deborah Estrin, Yakov Rekhter, and Steven Hotz. Scalable Inter-Domain Routing Architecture. In *SIGCOMM*, pages 40–52, 1992.

[FFF99] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On Power-law Relationships of the Internet Topology. In *SIGCOMM*, pages 251–262, 1999.

[FLP85] Michael J. Fischer, Nancy A. Lynch, and Michael S. Paterson. Impossibility of Distributed Consensus with One Faulty Process. In *Journal of the ACM*, volume 32(2), pages 374–382, April 1985.

[fni] Fault-Tolerant Networking through Intrusion Identification and Secure Compartments. `http://fniisc.nge.isi.edu`.

[GLAM95] J.J Garcia-Lunes-Aceves and S. Murthy. A Loop-Free Path-Finding Alogirthm: Specification, Verification and Complexity. In *Proceedings of the IEEE INFOCOM*, April 1995.

[GP01] T. Griffin and B. J. Premore. An Experimental Analysis of BGP Convergence Time. In *Proceedings of ICNP*, November 2001.

[GR00] L. Gao and J. Rexford. Stable Internet Routing Without Global Coordination. In *Proceedings of ACM SIGMETRICS*, June 2000.

[GSW99] T. Griffin, F. B. Shepherd, and G. Wilfong. Policy Disputes in Path Vector Protocols. In *Proceedings of ICNP*, November 1999.

[GW99] T. Griffin and G. Wilfong. A Analysis of BGP Convergence Properties. In *Proceedings of ACM SIGCOMM*, September 1999.

[GW00] T. Griffin and G. Wilfong. A Safe Path Vector Protocol. In *Proceedings of IEEE INFOCOMM*, March 2000.

[Haa01] J. Haas. Autonomous System Number Substitution on Egress. Internet Draft, Work in Progress, IETF, 2001.

[HB96] J. Hawkinson and T. Bates. Guidelines for creation, selection, and registration of an Autonomous System (AS). RFC 1930, 1996.

[HHW+01] J. Haas, S. Hares, S. Willis, J. Burruss, and J. Chu. Definitions of Managed Objects for the Fourth Version of Border Gateway Protocol (BGP-4). Internet Draft, draft-ietf-idr-bgp4-mib-08.txt, IETF, 2001.

[HM00] Sam Halabi and Danny McPherson. *Internet Routing Architectures, 2nd Edition*. Cisco Press, 2 edition, January 2000.

[Hui00] Christian Huitema. *Routing in the Internet*. Prentice-Hall, 2 edition, January 2000.

[Hum91] Pierre A. Humblet. Another Adaptive Distributed Shortest Path Algorithm. *IEEE Transactions on Communications*, 39(6):999–1003, 1991.

[Hus] G. Huston. BGP table statistics. `http://www.telstra.net/ops/bgp/as6447/bgp-multi-orgas.html`.

[Hus98] Geoff Huston. *ISP Survival Guide: Strategies for Running a Competitive ISP*. John Wiley and Sons, October 1998.

[Hus99] G. Huston. Interconnection, Peering, and Settlements, 1999. `http://www.potaroo.net/papers/inet99/peering.htm`.

[Hus00] G. Huston. Interconnection and Peering. *Broad Band Satellite*, November 2000. `http://www.potaroo.net/ispcolumn/2000-11-peering.html`.

[Hus01a] G. Huston. Analyzing the Internet's BGP Routing Table. *The Internet Protocol Journal*, 4(1), March 2001.

[Hus01b] G. Huston. Commentary on Inter-Domain Routing in the Internet. RFC 3221, IETF, December 2001.

[HW01] K. Houle and G. Weaver. Trends in Denial of Service Attack Technology. `http://www.cert.org/archive/pdf/DoS_trends.pdf`, October 2001.

[irr] Internet Routing Registry. `http:www.irr.net`.

[Jeo01] D. Jeon. Understanding DDOS Attack, Tools and Free Anti-tools with Recommendation, April 2001. `http://rr.sans.org/threats/understanding_ddos.php`.

[KA98] S. Kent and R. Atkinson. Security Architecture for the Internet Protocol. RFC 2401, IETF, November 1998.

[KLMS00] S. Kent, C. Lynn, J. Mikkelson, and K. Seo. Secure Border Gateway Protocol (S-BGP) – Real World Performance and Deployment Issues. In *Proc. of Network and Distributed System Security Symposium*, February 2000.

[KLS00] S. Kent, C. Lynn, and K. Seo. Secure Border Gateway Protocol. In *IEEE Journal of Selected Areas in Communications*, number 4, April 2000.

[LABJ00] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. Delayed Internet Routing Convergence. In *Proceedings of ACM SIGCOMM*, August 2000.

[LABJ01] Craig Labovitz, Abha Ahuja, Abhijit Bose, and Farnam Jahanian. Delayed Internet routing convergence. *IEEE/ACM Transactions on Networking*, 9(3):293–306, June 2001.

[LAJ99] C. Labovitz, A. Ahuja, and F. Jahanian. Experimental Study of Internet Stability and Wide-Area Network Failures. In *Proceedings of FTCS99*, June 1999.

[Lan99] T. Lane. Hidden Markov Models for Human/Computer Interface Modeling. In *IJCAI-99 Workshop on Learning about Users*, pages 35–44, 1999.

[LMJ97] Craig Labovitz, G. Robert Malan, and Farnam Jahanian. Internet routing instability. In *Proceedings of ACM SIGCOMM*, volume 27,4 of *Computer Communication Review*, pages 115–126, New York, September 14–18 1997. ACM Press.

[LMJ98a] Craig Labovitz, G. R. Malan, and Farnam Jahanian. Origins of internet routing instability. Technical Report CSE-TR-368-98, University of Michigan Department of Electrical Engineering and Computer Science, July 17, 1998.

[LMJ98b] Craig Labovitz, G. Robert Malan, and Farnam Jahanian. Internet routing instability. *IEEE/ACM Transactions on Networking*, 6(5):515–528, October 1998.

[Los00] Pete Loshin. *Big Book of Border Gateway Protocol (BGP) RFCs*. Morgan Kaufmann Publishers, June 2000.

[LPS82] Leslie Lamport, Marshall Pease, and Robert Shostak. The Byzantine Generals Problem. In *ACM Transactions on Programming Languages and Systems*, volume 4(3), pages 382–401, July 1982.

[LR89] K. Lougheed and Y. Rekhter. Border Gateway Protocol. RFC 1105, IETF, June 1989.

[LR90] K. Lougheed and Y. Rekhter. Border Gateway Protocol. RFC 1163, IETF, June 1990.

[LR91] K. Lougheed and Y. Rekhter. Border Gateway Protocol 3. RFC 1267, IETF, October 1991.

[LTG+92] T. Lunt, A. Tamaru, F. Gilham, R. Jagannathan, P. Neumann, H. Javitz, A. Valdes, and T. Garvey. A real-time intrusion detection expert system (IDES) - final technical report. Technical report, Computer Science Laboratory, SRI International, Menlo Park, California, 1992.

[Lun93] T. Lunt. Detecting Intruders in Computer Systems. In *Proceedings of the 1993 Conference on Auditing and Computer Technology*, 1993.

[LWVA01] C. Labovitz, R. Wattenhofer, S. Venkatachary, and A. Ahuja. The Impact of Internet Policy and Topology on Delayed Routing Convergence. In *Proceedings of IEEE INFOCOMM*, April 2001.

[Mal98] Gary Malkin. Routing Information Protocol Version 2. RFC 2453, SRI Network Information Center, November 1998.

[Mas00] Daniel Massey. *Fault Detection and Security in Routing Protocols*. PhD thesis, UCLA, 2000.

[Met96] B. Metcalfe. Netcom-cisco outage could foreshadow much bigger collapses ahead. *InfoWorld*, July 1996. `http://www.infoworld.com/cgi-bin/displayNew.pl?/metcalfe/bm070896.htm`.

[MGLA94] S. Murthy and J.J Garcia-Lunes-Aceves. A Loop-Free Alogirthm Based On Predecessor Information. In *Proceedings of IEEE ICCCN*, October 1994.

[MGLA95a] S. Murthy and J.J Garcia-Lunes-Aceves. Dynamics of a Loop-Free Path-Finding Algorithm. In *Proceedings of IEEE Globecom*, November 1995.

[MGLA95b] S. Murthy and J.J Garcia-Lunes-Aceves. A Routing Protocol for Packet Radio Networks. In *Proceedings of ACM Mobile Computing and Networking*, November 1995.

[Moo01] D. Moore. The Spread of the Code-Red Worm (CRv2), 2001. `http://www.caida.org/analysis/security/code-red/coderedv2_analysis.xml`.

[Moy94] J. Moy. OSPF Version 2. RFC 1583, IETF, March 1994.

[MR01] D. Massey and S. Rose. DNS Security Introduction and Requirements. Technical report, IETF, September 2001. Internet Draft draft-ietf-dnsext-dnssec-intro-01.txt.

[mrt] MRTD: The Multi-Threaded Routing Toolkit. `http://www.mrtd.net`.

[Mur02a] S. Murphy. BGP Security Protections. Internet Draft, Work in Progress, IETF, February 2002.

[Mur02b] S. Murphy. BGP Security Vulnerabilities Analysis. Internet Draft, Work in Progress, IETF, February 2002.

[nan97] 7007 explanation and apology, April 1997. http://www.cctec.com/maillists/nanog/historical/9704/msg00444.html.

[nan98a] AS8584 taking over the internet, April 1998. http://www.cctec.com/maillists/nanog/historical/9804/msg00047.html.

[nan98b] AS8584, April 1998. http://www.cctec.com/maillists/nanog/historical/9804/msg00048.html.

[nan98c] Re: Strange bgp announcement, November 1998. http://www.cctec.com/maillists/nanog/historical/9811/msg00275.html.

[nan01] C&W routing instability. Technical report, April 2001. http://www.cctec.com/maillists/nanog/historical/0104/msg00209.html.

[nla] National Laboratory for Applied Network Research. http://moat.nlanr.net/Routing/rawdata.

[pch] PCH.net. http://www.pch.net/documents/data/routing-tables/route-views.oregon-ix.%net.

[Per88] Radia Perlman. *Network Layer Protocols with Byzantine Robustness*. PhD thesis, MIT, August 1988.

[PZW+02] D. Pei, X. Zhao, L. Wang, D. Massey, A. Mankin, S. F. Wu, and L. Zhang. Improving BGP Convergence Through Consistency Assertions. In *IEEE INFOCOM*, June 2002.

[ris] RIPE Routing Information Service. `http://www.ripe.net/ripencc/pub-services/np/ris-index.html`.

[RL95] Y. Rekhter and T. Li. Border Gateway Protocol 4. RFC 1771, IETF, July 1995.

[rou] The Route Views Project. `http://www.routeviews.org`.

[SGLA96] B. Smith and J.J. Garcia-Luna-Aceves. Securing the Border Gateway Routing Protocol. In *Proceedings of Global Internet*, November 1996.

[ssf] The SSFNET Project. `http://www.ssfnet.org`.

[Ste99] John W. Stewart. *BGP4 Inter-Domain Routing in the Internet*. Addison-Wesley Pub Co, January 1999.

[tre02] Overview of Attack Trends, August 2002. `http://www.cert.org/archive/pdf/attack_trends.pdf`.

[VAMM99] C. Villamizar, C. Alaettinoglu, D. Meyer, and S. Murphy. Routing Policy System Security. RFC 2725, IETF, December 1999.

[VGE00] Kannan Varadhan, Ramesh Govindan, and Deborah Estrin. Persistent route oscillations in inter-domain routing. *Computer Networks*, 32(1):1–16, 2000.

[Wan00] Feiyi Wang. *Vulnerability Analysis, Intrusion Prevention and Detection for Link State Routing Protocols*. PhD thesis, NCSU, 2000.

[WZP+02] L. Wang, X. Zhao, D. Pei, D. Massey, A. Mankin, S. F. Wu, and L. Zhang. Protecting BGP Routers to Top Level DNS Servers, 2002. Submitted to ACM SIGCOMM Computer Communication Review.

[XDGLA97] Z. Xu, S. Dai, and J.J. Garcia-Luna-Aceves. A More Efficient Distance Vector Routing Algorithm. In *Proceedings of IEEE MILCOM*, November 1997.

[Yu] Jessica Yu. A Route-Filtering Model for Improving Global Internet Routing Robustness. `http://www.iops.org/Documents/routing.html`.

[Zak] R. Zakon. Hobbes' Internet Timeline v5.6. `http://www.zakon.org/robert/internet/timeline/#Growth`.

[ZMM+01] X. Zhao, A. Mankin, D. Massey, D. Pei, L. Wang, S. F. Wu, and L. Zhang. Validation of Multiple Origin ASes Conflicts through BGP Community Attribute. Internet Draft, draft-zhao-idr-moas-validation-00.txt, IETF, 2001.

[ZPW+01] X. Zhao, D. Pei, L. Wang, D. Massey, A. Mankin, S. F. Wu, and L. Zhang. An Analysis of BGP Multiple Origin AS (MOAS) Conflicts. In *ACM SIGCOMM Internet Measurement Workshop*, November 2001.

[ZPW+02] X. Zhao, D. Pei, L. Wang, D. Massey, A. Mankin, S. F. Wu, and L. Zhang. Detection of Invalid Routing Announcement in the Internet. In *The International Conference on Dependable Systems and Networks*, June 2002.