Information Technology

An Experimental Framework for BGP Security Evaluation

Ms Debbie Perouli: Purdue University, 47907-2107 West Lafayette, IN, U.S.A.
Tel: +1-765-496-9398, Fax: +1-765-494-0739, E-Mail: depe@purdue.edu
Debbie is a Ph.D. candidate in the Computer Science department at Purdue University, West Lafayette, IN, USA. She received her Bachelor's degree (5 year Diploma) in Electrical and Computer Engineering from the National Technical University of Athens (NTUA), Greece, in 2006. Her research interests include BGP routing, path algebras, network modeling and configuration.

Dr Olaf Maennel: Loughborough University, Computer Science, LE11 3TU UK
Tel: +44 (0)1509 222681, E-Mail: o.m.maennel@lboro.ac.uk
Olaf is a lecturer at Loughborough University in the United Kingdom since September 2009. Before that he was at Telekom Innovation Laboratories and at the School of Mathematical Science at the University of Adelaide in South Australia. He graduated from the Technical University in München (Germany) in 2005. His research interests are network security, routing, active measurements, and next generation internet technology.

Dr Iain Phillips: Loughborough University, Computer Science, LE11 3TU UK
Tel: +44 (0)1509 222681, E-Mail: i.w.phillips@lboro.ac.uk
Iain is a Senior Lecturer in Computer Science at Loughborough Univeristy, UK. He has a Ph.D. and B.Sc from Computer Science at Manchester in the UK. His research interests are Network Architectures and Performance Measurement, including both the Internet and Wireless Networks.

Dr Sonia Fahmy: Purdue University, 47907-2107 West Lafayette, IN, USA
Tel: +1-765-494-6183, Fax: +1-765-494-0739, E-Mail: fahmy@cs.purdue.edu
Sonia is a professor of Computer Science at Purdue University, USA. She received her PhD degree from the Ohio State University in 1999. Her current research interests include network testbeds, network security, and wireless sensor networks.

Mr Randy Bush: Internet Initiative Japan, Tokyo, Japan
E-Mail: randy@psg.com
Randy is a Research Fellow and network operator at Internet Initiative Japan, Japan's first commercial ISP and a visiting Professor at Loughborough. He specializes in network measurement especially routing protocols, network security, and IPv6 deployment. He has been heavily involved in transferring Internet technologies to developing economies for over 20 years.

Mr Rob Austein: Dragon Research Labs, USA
E-Mail: sra@hactrn.net
Rob is a Principal Research Scientist at Dragon Research Labs. He has been active in Internet protocol specification and development since the mid-1980s. His current research interests include network security and loosely consistent distributed databases.

Keywords: Internet inter-domain routing; Border Gateway Protocol (BGP); security; testbeds

Schlagworte: Internet inter-domain routing; Border Gateway Protokoll (BGP); Sicherheit; Testbett

MS-ID:
Heft: 53/* (1886)

December 6, 2012

Abstract

Internet routing is based on implicit trust assumptions. Given the critical importance of the Internet and the increasing security threats, such simple trust relationships are no longer sufficient. Over the past decade, significant research has been devoted to securing the Internet routing system.

The Internet Engineering Task Force (IETF) is well along in the process of standardizing routing security enhancements (Secure Inter-Domain Routing—SIDR, Keying and Authentication for Routing Protocols–KARP, etc.). However, the research challenges are not over: not only do these new protocols need to be tested for protocol conformance and interoperability, they also need to be evaluated both for their security properties and scaling performance.

The purpose of this paper is two-fold: we outline the main security challenges in inter-domain routing and argue that research in this area has barely begun; and we take a closer look at a production implementation of one component and evaluate it at a fairly large scale. We discuss the difficulties we experienced and lessons learned; we also present some initial results.

Zusammenfassung

Als die Protokolle für das Internet entwickelt wurden, hatte man sich um Sicherheitsfragen nicht wirklich gekümmert. Heute jedoch hat das Internet an Wichtigkeit zugenommen und ganz besonders im Bereich Routing reicht das einst noch ausreichende Vertrauen zwischen den Internet-Anbietern nicht mehr aus. Vor mehr als 10 Jahren wurden die Sicherheitsprobleme erkannt und seit dem haben die Arbeiten angefangen, das Routing System des Internet sicherer zu machen.

Die Internet Engineering Task Force (IETF) ist nun endlich dabei, die Sicherheitsstandards für das Routing im Internet festzulegen (siehe unter anderen die Arbeiten von SIDR in der IETF—Secure Inter-Domain Routing und KARP—Keying and Authentication for Routing Protocols). Allerdings kann nicht davon gesprochen werden, dass die Herausforderungen bewältigt wären: diese neuen Protokolle müssen auf Stimmigkeit und fehlerfreies Zusammenarbeiten getestet werden, aber man sollte sie auch auf ihre Sicherheitseigenschaften und Skalierbarkeit überprüfen.

Das Ziel dieser Forschungsarbeit ist zum einen, dass wir den Stand und die Herausforderungen im Bereich Inter-domain Routing Sicherheit diskutieren und zeigen, dass die Forschung auf diesem Gebiet gerade erst begonnen hat; zum anderen werfen wir einen kritischen Blick auf die ersten Implementierungen und evaluieren sie in einem größeren Labor. Wir erläutern Schwierigkeiten und was wir bei unseren Versuchen gelernt haben, darüberhinaus präsentieren wir einige erste Messergebnisse.

1 Introduction

As the Internet grows and offers new services to users, engineering its components and ensuring its security have become increasingly challenging. The Internet started as a network of trusted parties, and external threats were not considered in its initial design [1]. Since then, security vulnerabilities have appeared in real networks and their impact has propagated over the entire Internet; prominent examples include [2, 3]. Experts have demonstrated [4] that such security loopholes can be exploited, leading to severe intended or inadvertent malfunction. Miscreants have also demonstrated several weaknesses.

Central to the operation of the Internet are its routing protocols, which discover the paths that traffic should follow. At the inter-domain level, the Border Gateway Protocol (BGP) [5] is the global routing protocol which interconnects networks belonging to different Internet Service Providers (ISPs), *i.e.*, administrative domains. Unfortunately, BGP suffers from a range of security vulnerabilities [6,7]. These include misorigination of prefixes (known as "route hijacks"), protocol attacks [8], and accidental misconfigurations [9].

The research community has responded with several promising ideas to secure BGP [10,11]—some more complete or practical than others. As a result of these and other efforts, the Internet Engineering Task Force (IETF) Secure Inter-Domain Routing (SIDR) working group has proposed new standards [12] leveraging an X.509 certificate based Resource Public Key Infrastructure (RPKI) [13]. Initial implementations of these standards are available (*e.g.*, an early production version of an open-source RPKI implementation [14]) and are being tested by ISPs.

It is critical that the architecture, implementation, and operational implications of any proposed framework be thoroughly evaluated before it is widely deployed in the Internet. The research community should investigate which security issues a proposed solution addresses, how well they are addressed, and how critical the remaining vulnerabilities are [15]. Before introducing a new system, operators first need to understand the consequences of its deployment. One aspect of concern is *scalability*, as a good idea may perform well in a small test-lab but fail at Internet scale. Our aim is to develop a methodology suitable for evaluating implementations with respect to their adherence to specifications, their interoperability, and their degree of scalability.

In this paper, we seek to address this question:

> *How can we determine if new BGP security protocols are ready for large-scale, Internet-wide deployment?*

Both the complexity of the proposed solutions as well as the expected deployment scale suggest that simple protocol compliance tests are insufficient: operational deployment and scalability tests are critical. Furthermore, system soundness during day-to-day operations must be demonstrated.

We argue for the importance of *high-level abstractions* to enable *auto-generation of experimental topologies*. We believe that such generated topologies deployed on large virtualized networks provide the means to experimentally evaluate complex systems and protocols at scale. This paper is an extended version of our workshop paper [16], which focused mainly on abstract network configurations. Here, we present some preliminary results and immediate lessons learned.

Our contributions include tools to generate test scenarios, and a virtualized large-scale platform to conduct tests. We develop an abstract representation of the entities and interactions involved in the RPKI specifications. We also perform initial experiments with auto-configured virtual topologies which are not yet at the target scale but have already helped uncover problems in the RPKI implementation.

2 Securing BGP

BGP is a path-vector protocol that announces and withdraws reachability information. The protocol determines the most appropriate paths based on the information propagated.

Securing BGP from vulnerabilities involving misuse of the protocol itself decomposes into a number of subproblems, including (1) validating that the originating provider had the right to announce an address space, and (2) validating that a received announcement actually traversed the path claimed. Path validation defends against path diversion. This level of security does not protect against route leaks [15]. Although no formal definition exists, route leaks are understood as cases in which routes are propagated with their authentic origins, but are misdirected through unexpected intermediaries [17]. Finally, there is no BGP or other protocol mechanism in place that ensures that traffic will actually follow the selected routing path [7,18].

In the following, we briefly review the main solutions that the IETF SIDR working group is proposing to address most of these security issues.

2.1 RPKI

The RPKI is an X.509 certificate-based hierarchy congruent with the Internet IP address and Autonomous System (AS) number allocation administration, the IANA, RIRS, ISPs, etc. It is the substrate on which origin and path validation are based. The RPKI is currently deployed in all five of the administrative regions of the Internet.

Certification: The RPKI exchanges and publishes X.509 certificates which are used to attest to who owns what address space and AS numbers. In the setup phase, identities and keys are exchanged to establish the cryptographic relationships between "parent" and "child." This relationship hierarchy follows the allocation hierarchy, *e.g.*, IANA gives address space to ARIN, ARIN to ISPs, ISPs to customers.

The RPKI certificates do not specify which ISP should announce a particular address block in BGP. The owner of the address space specifies which AS or ASes may announce the space by issuing a Route Origin Authorization (ROA) [19] which associates a prefix with an authorized originating AS number. In order to publish ROAs, EE-certificates (End Entity) are generated which sign the corresponding ROAs.

Publication hierarchy: Certificates, Certificate Revocation Lists (CRLs), Manifests, and signed objects such as ROAs are stored in a repository system. The purpose of this system is to provide the information to *relying parties* who pull data from the system at the frequency they choose. The repository system comprises multiple databases which are distributed among the certifying authorities. It is the choice of the network operator to specify whether to self-publish (run their own *publication repository*) or delegate. In the delegated case, if an AS is *hosted by* another entity such as a Regional Internet Registry (RIR), then the RIR publishes certificates and other objects for that AS.

Verification: The relying parties, *e.g.*, ISPs, maintain local caches which fetch information from the distributed repository system, verify it, and store the validated information. The caches are typically located at Points of Presence (PoPs), close to the routers. Tools such as *rcynic* [14] fetch the cryptographic objects.

Rcynic collects required information (such as certificates and ROAs) from repositories. Suppose that a certificate B is issued and signed by the parent's certificate A. This parent certificate contains a Subject Information Access (SIA), which is a URI (Uniform Resource Identifier) pointing to the publication point of certificate B. All certificates are linked in this way. The Authority Information Access (AIA) is the reverse and links certificate B to the parent's certificate A.

A cache may fetch information from the global RPKI infrastructure (*i.e.*, all the publication points in the Internet), in which case we call it a *gatherer*, or it might fetch information from another cache. In both cases, gatherers and caches independently verify the cryptographic information.

2.2 Origin Validation

BGP origin validation [19] ascertains the mapping of prefixes to origin AS numbers in each BGP UPDATE message received at a router in the Internet. The AS-path attribute within the UPDATE message contains a list of ASes that the BGP message has traversed (most recent first). The last AS on the path is the origin.

For scalability reasons, routers participating in BGP origin validation are not expected to undertake cryptographic operations. The correctness of prefix to origin AS mapping is validated on caches using the information in ROAs. The list of mappings is then transferred to the router [20].

Once a BGP UPDATE arrives at a router, it can be checked against this list of prefix to AS number mappings. If the last AS on the received AS path is contained in that list (for that prefix), then the UPDATE is said to be *valid*. In case the last AS in that list does not match the AS stated in the mapping tables, the UPDATE is *invalid*. Finally, if there is no entry in the tables, the status of the UPDATE is *unknown*.

Although such a solution addresses significant problems caused by misconfigurations, it does not protect against many deliberate attacks. An attacker can still announce any path as long as it ends with the valid AS. The announcement does not need to actually originate from the valid AS, the attacker can just fake the origin AS. Since there are no cryptographic certificates on the router, a router cannot detect this type of attack.

2.3 AS Path Validation

As with the origin of a prefix, BGP does not provide a mechanism for the recipient of an UPDATE to verify the AS path contained in the route announcement. Paths, then, are open to intended or accidental manipulation. In addition to this vulnerability, since BGP messages are not signed, they are also open to malicious spoofing.

BGPsec [21,22] is a protocol that satisfies the requirements for BGP path validation [23]. An eBGP speaker is able to sign an UPDATE. As the message traverses a sequence of ASes, each eBGP speaker adds its own public key hash and digital signature to the BGPsec attribute sequence. The SIDR work on implementing the BGPsec solution is at an earlier stage, compared to the AS origin validation.

BGPsec does not offer a solution to route leaks that follow legitimate paths. For example, it has been shown that Man In The Middle (MITM) attacks can still be launched in a BGPsec-enabled environment [15, 17].

3 Challenges

As outlined in the previous section, the IETF proposals do not yet completely address BGP security concerns. However, the RPKI already provides a solution to misconfiguration problems. We believe that the RPKI provides a solid framework to incorporate techniques that will address further security issues. There has been significant effort in creating prototype implementations of the proposed RPKI architecture. In the remainder of this paper, we will evaluate a publicly available implementation [14] that is also being tested by the RIRs and some large ISPs.

A difficult challenge is the lack of a testing methodology and tools that allow security experts, router vendors, and ISPs to explore the system at a large scale before real-world deployment. Such tools need to run real vendor code and infrastructure software that is used in the Network Operation Centers (NOCs).

The value of test-labs and tools that allow high fidelity experimentation at large scales has been widely recognized [24, 25]. Several test-labs are available today, each with its own benefits and limitations. DETER [24] is a quarantined testbed aimed at security experiments. DETER includes a set of general management tools, *e.g.*, [26, 27]. GENI [25] is a federation that includes PlanetLab [28], ProtoGENI (based on Emulab [29]) and DETER. While some parts of DETER and ProtoGENI are isolated, parts of GENI like PlanetLab use Internet links, so they cannot be used for reproducible experiments. StarBED [30], a compute-cluster in Japan, allows experimenters to perform large-scale evaluation tests.

It is non-trivial to configure a test-lab with thousands of nodes involving interactions of multiple ASes, even if that test-lab is run as a set of virtual machines (VMs) on a compute cluster. The reason is that many separate but coordinated device configurations need to be created—a time consuming, repetitive and error-prone task [27].

To allow repeated execution of BGP security experiments in a fast and flexible manner, an isolated, fully functional emulated network should be automatically deployed. This entails auto-generation of large parts of the BGP security components as well as the network infrastructure. We need abstract descriptions to model multi-ISP topologies and reasonable defaults to generate the detailed device (router and server) configurations. These abstractions enable the creation of networks of varied complexity with minimum specification effort.

Unfortunately, BGP security evaluation poses unique requirements on a test-lab. For streamlined RPKI evaluation, we found that we need more flexibility and additional tools than those provided by today's test-labs. Our work thus augments existing test environments with custom tools that specifically target BGP security experiments.

We extend abstractions in AutoNetkit [31] – a configuration generation tool for complex network emulations. Our abstractions are discussed in detail in [16]. Overall, we strive towards simplicity, but we also need to capture the structures of the SIDR framework including: (1) a test-lab topology, which we express as a graph with node and edge attributes indicating different functions (*e.g.*, cache, router); (2) the publication hierarchy, which includes the RPKI CAs (Certification Authorities); (3) the relying party cache servers, including the rtr-protocol; and (4) the router BGP configurations.

4 Methodology

Our goal is to enable easy creation and deployment of test scenarios. Therefore, our toolkit only has four phases: (1) download the tool and accompanying semi-configured Linux images, *e.g.*, the RPKI software [14]; (2) specify the abstractions; (3) compile to configurations and deployment of VMs; (4) run experiments.

We aim at recreating "semi-realistic" behavior, which is crucial for the security evaluation of the proposed SIDR architectures. We auto-generate what we anticipate "good players" would do. We do not aim to create attacks, or what "evil players" may be doing. We leave this to security experts who use our tool to facilitate a lab testing. However, we believe there is great value in bootstrapping the initial deployment process with semi-realistic configurations.

We call a static, generated setup a *scenario*. A virtual topology is deployed and the related protocols, like those involved in routing and RPKI, propagate information in the network. In contrast to a scenario, an *event* can be an addition, modification or revocation of a ROA. Our tool automatically generates events with a repeatable pattern.

Note that we generate reasonable defaults whenever the user is not specific in his/her input to the tool. For example, IP addresses are automatically allocated. The corresponding configuration entries are created for each VM according to the specified platform. This allows the experimenters to focus on their tests instead of configuration details.

An example: Figure 1 shows a configuration example that specifies, configures and deploys a multi-AS topology. We aim to express most of the configuration visually in an input graph file. Therefore, the figure shows the complete configuration input. Only a single configuration file, which details access to our compute-server, is omitted. Node properties in the graphml input are used to specify properties such as to which AS the node belongs, or if it is an RPKI CA server (*i.e.*, runs *rpkid*, the CA daemon speaking the Up/Down Protocol with other RPKI servers). In this example, *rN* (where $N = 1..6$) denotes a router; *rpki-X* an RPKI CA server; and *cacheM* (where $M = 1..4$) a cache server.

4.1 Implementation

For specifying the test-lab topology, we aim at roughly the same level of abstraction that one would use when drawing on a whiteboard. The user describes nodes and edges using graphml properties. We define *physical* links and *logical* relationships between nodes, set by link properties in the graphml. In Figure 1, physical links are drawn as solid lines, while all logical relationships are represented by dashed lines.

Our tool deploys the RPKI configurations to Linux VMs and router configurations to Junosphere. We use StarBED [30] as the deployment platform. Each StarBED machine hosts around 20 Virtual GNU/Linux boxes (using KVM as hypervisor). Each VM runs a specific part of the RPKI infrastructure. All Linux VMs are configured using two disk images: (1) a modified installation with all required software, and (2) an image with configuration files, which is created during the compilation phase. We focus on integrating real router vendor code as this allows us to evaluate BGPsec [32] implementations as vendors make them available.

We also collect CPU, memory, load, and other statistics on both the hosts and the guest machines (using *rrdtool*), and monitor our VMs using NAGIOS. The VM information can be carefully inspected when heavy load is observed in order to explain the results. A virtual layer 2 environment is created (using Openvswitch) which allows all the Linux VMs to communicate.

A key challenge is to ensure that the effects of the hypervisor environment on the experimental results can be understood. Clearly, a virtualization hypervisor will distort time-dependent characteristics. Experiments requiring strict timing (of the order of seconds or less) or high-throughput cannot be accurately tested. However, for the scaling and security measurements currently contemplated by vendors and operators, this should not be a major problem. We believe that our approach is valid when the required accuracy of time measurements is of the order of minutes. Comparing different implementations of the same protocol or monitoring the propagation of cache updates in large-scale deployments are the kinds of experiments we are targeting. By monitoring load statistics on the host machines we can ensure, to a reasonable extent, that protocol behavior is not affected by the hypervisor environment.

5 Experiment: Repository Structure

In this section, we discuss the lessons we learned from conducting initial experiments with the RPKI system. We focus on propagation time through the system, *i.e.*, the time it takes from the publication of an object until all caches and/or routers receive it. As a bonus, while setting up an experiment with about 800 Linux virtual machines, we found a number of implementation optimizations.

5.1 Topology

The virtual topology consists of tier-1, tier-2 and tier-3 ISPs. Each ISP has four customer ASes whose RPKI data are hosted by that ISP. ISPs that belong to a different tier have a different number of caches, including the gatherers. Some of the tier-3 ISPs also differ in that they do not run their own RPKI CA.

Clearly, the size of the experimental topology does not represent an Internet-scale environment. However, our auto-configuration tools are particularly designed to make scaling up an experiment easier.

5.2 Hierarchical vs. Flat Publication

As stated in [33], a hierarchical organization of repositories is expected to benefit the performance of relying party gathering data as opposed to a flat organization. Therefore, publishing parties should implement hierarchical directory structures. The performance difference is due to the number of *rsync* connections needed in each case [34].

In a hierarchical structure, the filesystem hierarchy within the repository follows the certificate hierarchy. Suppose that Alice issues a certificate for Bob, where certificates and signed objects of both are hosted in the same repository. In a hierarchical structure, the publication points would look like:

```
rsync://example.org/rpki/Alice/
rsync://example.org/rpki/Alice/Bob/
```

In a flat structure, the certificate hierarchy is ignored, so the publication points will be:

```
rsync://example.org/rpki/Alice/
rsync://example.org/rpki/Bob/
```

In the first case, a single *rsync* connection retrieves both Alice's and Bob's objects, while two *rsync* connections are needed in the second case. Considering the fact that most certifying authorities will have more than one child, the number of *rsync* connections is dramatically increased in a flat structure.

This was confirmed in the real Internet when the RIPE NCC converted from a flat to a hierarchical publication structure. Figure 2[1] illustrates the synchronization time reduction due to the repository structure change. The y-axis shows the total time needed to synchronize a local gatherer against RIPE's publication site for a particular *rcynic* run. When the structure turned hierarchical, sync time dropped by a factor of at least 35: from approximately 3700 down to 100 seconds.

Figure 3 depicts results from our emulations. The y-axis shows the propagation time from the publication points to the gatherers. In this particular experiment, each gatherer synchronizes with the repository every 60 minutes, though not all synchronize at the same time.

[1] Reported by `http://www.hactrn.net/opaque/rcynic/rpki.ripe.net_month_svg.html`, 2012-11-09

Therefore, the ideal behavior would be that all gatherers receive the published objects by the end of a 60 minute period. The fastest propagation times (blue line on the left in Figure 3) is for hierarchical, while the slowest (black line on the right) appears for flat publication. Note that this graph is indicative of performance of one implementation of the RPKI code, and not necessarily the performance of other implementations.

5.3 Lessons Learned

Our experiments used parts of the RPKI code in ways that developers did not anticipate (though still legitimate), and with larger test cases. This allowed us to identify new problems.

Adding new entities to the system was not expected to be a bulk operation as would occur in the start-up of a new CA loading legacy data. Each new entity needs changes in the database of certificates and signed objects. In our scripts, we tried adding more than one thousand entities at a time, which caused severe CPU overload, in part due to the CPU overhead of generating thousands of 2048-bit RSA keys, while trying to process a long list of change requests.

Furthermore, our scripts included cases in which there were internal dependencies among the entities being configured in a single batch, *e.g.*, cases where a parent, its children, and its grandchildren were all configured in a single batch. Internally, this required the ancestors to be fully configured before it was possible to configure their descendants. One code modification that significantly improved performance was to switch to a more sophisticated approach to internal task scheduling within *rpkid* to avoid resource starvation and configuration protocol timeouts.

Our experiments also revealed that the default values of some parameters were not appropriate for all cases. For the expected load of entity additions that an operator will have to go through on a daily basis, parameters related to *cron* cycle runs were set correctly. However, the same parameters were too small to accommodate the initial loading of a large amount of existing data. This observation also lead to modifying the code with appropriate values.

Overall, our experience demonstrates that some problems only arise when the code is executed at large scale. We therefore believe that conducting large-scale experiments will continue to contribute to code quality.

6 Conclusion

We have discussed the challenges the research community faces in securing BGP, the inter-domain protocol of the Internet. There are important security concerns that are not yet addressed by IETF proposals and there is a lack of tools to evaluate the existing solutions, such as the RPKI, at a large scale.

We propose an abstract representation that allows us to experiment with the SIDR proposals to secure BGP. This abstraction is necessary to bring configuration simplicity to controlled test-labs and thence to evaluate complex protocol and system interactions. The system has already had some success in finding problems with scalability in the RPKI software, which have led to code improvements. We plan to extend our experiments with larger topologies that resemble more of the characteristics of the Internet.

Acknowledgments

We would like to thank the StarBED crew for their generous and continued support, Keiichi Shima (IIJ Innovation Institute), Graeme Fowler (Loughborough University IT Services) and Joel Obstfeld (Cisco Systems). This work is partially funded by a gift from Cisco grant #2011-89493.

References

[1] D. Clark, "The design philosophy of the darpa internet protocols," in *Symposium proceedings on Communications architectures and protocols*, ser. SIGCOMM '88. New York, NY, USA: ACM, 1988, pp. 106–114. [Online]. Available: http://doi.acm.org/10.1145/52324.52336

[2] M. A. Brown, "Pakistan hijacks YouTube," *Renesys blog*, February 2008, http://www.renesys.com/blog/2008/02/pakistan_hijacks_youtube_1.shtml.

[3] R. Hiran, N. Carlsson, and P. Gill, "Characterizing Large-scale Routing Anomalies: A Case Study of the China Telecom Incident," in *Proc. ACM Passive and Active Measurement Conference*, 2013.

[4] A. Kapela and A. Pilosov, "Stealing the Internet - A Routed, Wide-area, Man in the Middle Attack," 2008. [Online]. Available: http://defcon.org/html/defcon-16/dc-16-speakers.html#Kapela

[5] Y. Rekhter, T. Li, and S. Hares, "A Border Gateway Protocol 4 (BGP-4)," 2006, RFC 4271.

[6] G. Huston, M. Rossi, and G. Armitage, "Securing BGP - A Literature Survey," *Communications Surveys Tutorials, IEEE*, vol. 13, no. 2, pp. 199 –222, 2011.

[7] K. Butler, T. R. Farley, P. McDaniel, and J. Rexford, "A Survey of BGP Security Issues and Solutions," *Proceedings of the IEEE*, vol. 98, no. 1, pp. 100–122, January 2010.

[8] O. Nordström and C. Dovrolis, “Beware of BGP attacks,” *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 2, pp. 1–8, April 2004. [Online]. Available: http://doi.acm.org/10.1145/997150.997152

[9] D. Wetherall, R. Mahajan, and T. Anderson, “Understanding BGP misconfigurations,” in *Proc. ACM SIGCOMM*, 2002.

[10] S. Kent, C. Lynn, and K. Seo, “Secure Border Gateway Protocol (S-BGP),” *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 4, pp. 582 –592, 2000.

[11] R. White, “Securing BGP through secure origin BGP (soBGP),” *Business communication review*, vol. 33, no. 5, pp. 47–53, 2003.

[12] IETF Working Group, “Secure Inter-Domain Routing (sidr).” [Online]. Available: http://datatracker.ietf.org/wg/sidr/

[13] M. Lepinski and S. Kent, “An Infrastructure to Support Secure Internet Routing,” 2012, RFC 6480.

[14] Rob Austein, Dragon Research Lab. [Online]. Available: http://rpki.net/

[15] S. Goldberg, M. Schapira, P. Hummon, and J. Rexford, “How secure are secure interdomain routing protocols,” *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 4, pp. 87–98, August 2010. [Online]. Available: http://doi.acm.org/10.1145/1851275.1851195

[16] O. Maennel, I. Phillips, D. Perouli, R. Bush, R. Austein, and A. Jaboldinov, “Towards a framework for evaluating BGP security,” in *Proceedings of the 5th Workshop on Cyber Security Experimentation and Test*, Bellevue, WA, Aug 2012.

[17] D. McPherson, S. Amante, and E. Osterweil, “Route Leaks & MITM Attacks Against BGPSEC,” IETF Internet Draft, November 2012.

[18] S. Goldberg, S. Halevi, A. D. Jaggard, V. Ramachandran, and R. N. Wright, “Rationality and traffic attraction: Incentives for honest path announcements in BGP,” in *Proceedings of ACM SIGCOMM*, ser. SIGCOMM ’08. New York, NY, USA: ACM, 2008, pp. 267–278. [Online]. Available: http://doi.acm.org/10.1145/1402958.1402989

[19] G. Huston and G. Michaelson, “Validation of Route Origination Using the Resource Certificate Public Key Infrastructure (PKI) and Route Origin Authorizations (ROAs),” 2012, RFC 6483.

[20] P. Mohapatra, J. Scudder, D. Ward, R. Bush, and R. Austein, “BGP Prefix Origin Validation,” IETF Internet Draft, October 2012.

[21] Geoff Huston and Randy Bush, “Securing BGP and SIDR.” [Online]. Available: http://isoc.org/wp/ietfjournal/?cat=5328

[22] E. M. Lepinski, “BGPSEC Protocol Specification,” IETF Internet Draft, October 2012.

[23] S. Bellovin, R. Bush, and D. Ward, “Security Requirements for BGP Path Validation,” IETF Internet Draft, October 2012.

[24] J. Mirkovic, T. Benzel, T. Faber, R. Braden, J. Wroclawski, and S. Schwab, “The DETER project: Advancing the science of cyber security experimentation and test,” in *IEEE Technologies for Homeland Security*, nov. 2010, pp. 1 –7.

[25] C. Elliott and A. Falk, “An update on the GENI project,” *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 3, June 2009.

[26] J. Mirkovic, S. Wei, A. Hussain, B. Wilson, R. Thomas, S. Schwab, S. Fahmy, R. Chertov, and P. Reiher, “DDoS benchmarks and experimenter’s workbench for the DETER testbed,” in *Proceedings of TridentCom*, May 2007.

[27] R. Chertov, S. Fahmy, P. Kumar, D. Bettis, A. Khreishah, and N. B. Shroff, “Topology generation, instrumentation, and experimental control tools for emulation testbeds,” in *Proceedings of the DETER Community Workshop on Cyber Security Experimentation*, Arlington, Virginia, June 15-16 2006.

[28] “PlanetLab,” http://www.planet-lab.org/.

[29] M. Hibler, R. Ricci, L. Stoller, J. Duerig, S. Guruprasad, T. Stack, K. Webb, and J. Lepreau, “Large-scale virtualization in the emulab network testbed,” in *USENIX 2008 Annual Technical Conference on Annual Technical Conference*, ser. ATC’08. Berkeley, CA, USA: USENIX Association, 2008, pp. 113–128. [Online]. Available: http://dl.acm.org/citation.cfm?id=1404014.1404023

[30] “StarBED Project,” http://www.starbed.org/.

[31] H. Nguyen, M. Roughan, S. Knight, N. Falkner, R. Bush, and O. Maennel, “How to build complex, large-scale emulated networks,” in *TridentCom*, Berlin, Germany, May 2010.

[32] M. Lepinski and S. Turner, “An Overview of BGPSEC,” 2011, draft-ietf-sidr-bgpsec-overview-01.

[33] R. Bush, “RPKI-Based Origin Validation Operation,” IETF Internet Draft, August 2012.

[34] R. Austein, IETF Mail Archive, March 2012, http://www.ietf.org/mail-archive/web/sidr/current/msg04407.html.

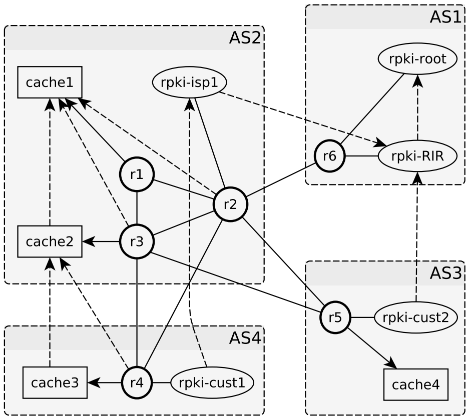

Figure 1: Example input based on our abstractions.

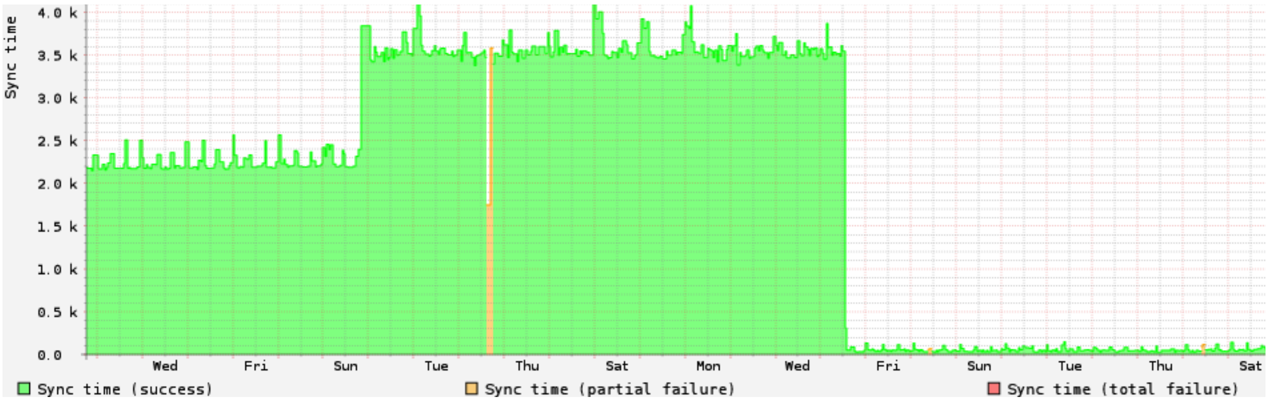

Figure 2: Synchronization times (in seconds) for RIPE NCC. On November 8, 2012 (Thursday) RIPE switched from flat to hierarchical publication repository structure which resulted in reduced sync times.

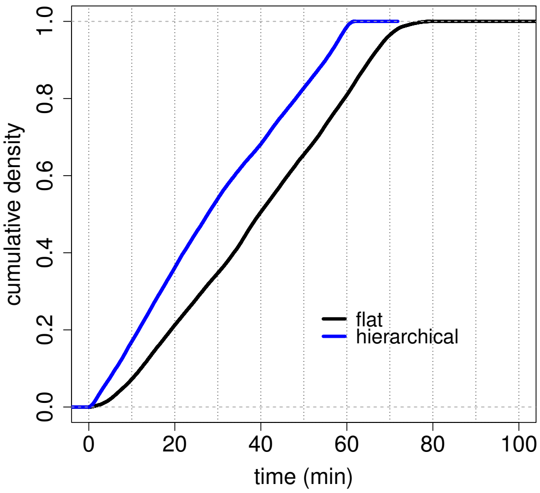

Figure 3: Propagation times for hierarchical (blue line on the left) and flat (black line on the right) publication.