Testing Large Scale BGP Security in Replayable Network Environments

Patrick McDaniel and Kevin Butler
Systems and Internet Infrastructure Security Laboratory
Department of Computer Science and Engineering
Pennsylvania State University

Abstract— **Understanding the operation of BGP and providing for its security is essential to the well-being of the Internet. To date, however, simulating every autonomous system comprising the Internet, in order to test proposals and security solutions, has been infeasible. We have developed *lseb*, a large scale BGP simulator, that generates realistic network topologies from routing data, and provides the ability to replay network events from any point in the past, and allows what-if scenarios such as simulated attacks or defense mechanisms, to test the resilience of the critical network infrastructure. We describe topology extraction tools that we have developed and the design and implementation of *lseb*. We also discuss visualization tools that allow a graphical topology representation and provide an example of an attack scenario that can be modeled with *lseb*.**

I. Introduction

The Border Gateway Protocol is the *de facto* interdomain routing protocol used in the Internet. Understanding the behavior and dynamics of BGP is essential to ensure the Internet's continued operation. Simulating the operation of BGP is difficult, however, because of the large scale it encompasses. There are over 22,000 autonomous systems (ASes) that comprise the current Internet, and BGP is responsible for all of the routing between these networks. As a result, simulating every facet of routing behavior rapidly becomes infeasible. The main drawback of existing network simulators is that they were not created with a goal of simulating the whole Internet, but rather of replicating detailed events in a small network setting. They mostly lack a realistic Internet model [14], [21] and simulate traffic at too fine a granularity, making the simulation prohibitively expensive [12], [11], [18], [16]. Many popular simulators run on a single node, which prevents large-scale simulation [1], while others run a distributed simulation [2], [3], [4] but to simulate at a reasonable speed, they require powerful and specialized clusters, which are not available to all researchers. In this paper, we propose a large-scale external BGP simulator, or *lseb*, that allows a full, Internet-wide simulation of BGP events. We use routing data from the Route Views data repository [13] to generate a realistic Internet topology and to simulate and replay actual routing events. *lseb* is able to perform these large scale simulations by eliding unnecessary data and can operate on commodity hardware over a large distributed system, such as the DETER testbed. We have made the source code available for use and further extension by researchers. The code can be found at `http://siis.cse.psu.edu/tools.html`.

II. Goals

We developed the *lseb* simulator in response to the needs of the BGP research community. Foremost with *lseb* is the ability to present a large-scale, realistic simulation of BGP operating across the entire Internet. With this infrastructure in place, and through the use of real routing data, we can create simulations that are reflective of real events with real topologies that reflect the state of the current Internet. Of even greater interest is the ability to use historical data to recreate the state of BGP in the Internet at a given time. Thus, we can have access to a "way-back" machine, where we can examine various what-if scenarios by modifying BGP behavior or injecting faults into, for example, one or more ASes. Thus, we can test the network infrastructures at critical junctures in time, such as during major power outages or under attack scenarios, and determine resiliency when attacks such as worm propagation, denial of service attacks or attacks against BGP are launched against the Internet. For example, the Internet infrastructure was severely stressed by the Code Red worm outbreak, which affected BGP convergence. By replaying the state of the network during the heart of the propagation period, we can simulate what would have transpired if an adversary had launched a link-cutting attack or a BGP-specific attack, such as prefix hijacking, during this period, and determine the ramifications on ASes throughout the Internet. These models will also be useful for determining the effectiveness of security mechanisms and validating results generated by us [6], [8], [17], [5] and others [10], [15], [20], [9]. For example, we can simulate the global adoption of schemes such as S-BGP [10] and soBGP [15], and see how optimizations such as SPV [9], signature amortization [20], or data-driven cryptographic constructions for origin authentication [6] and path authentication [5] scale when all 22,000 ASes that comprise the Internet are modeled.

III. Simulator Design and Implementation

Our design for providing large-scale Internet simulation is contingent on a series of processes that transform data from route repositories into a form that can be easily parsed for simulation. Raw routing data can be obtained from repositories such as Route Views or the RIPE RIS database [7]. With our *bgprv* tool suite (described elsewhere), we can filter and process routing data, eliding spurious information and transforming it into a easily parseable data that can be

Command	Description
AS	Usage: *node* `AS` *number*. Links an AS to a particular simulation node.
PEER	Usage: *node AS* `PEER` *AS-peer*. Creates a link between two ASes that are BGP neighbors.
PREFIX	Usage: *node* `PREFIX` *AS prefix*. Links a prefix with a given AS.

TABLE I

LIST OF COMMANDS USED BY THE TOPOLOGY GENERATOR.

Command	Description
START	Begins the simulation.
ADD	Usage: `ADD` *AS prefix*. Adds a specified prefix to the given AS.
DROP	Usage: `DROP` *AS prefix*. Removes the specified prefix from a given AS.
FAIL	Usage: `FAIL` *AS AS*. Causes a link failure between two connected ASes.
RECOVER	Usage: `RECOVER` *AS AS*. Recovers a link between two ASes from link failure.
DUMP	Dumps a copy of the routing tables of each AS.
SLEEP	Usage: `SLEEP` *time*. Pauses the master simulation thread for the specified time.
STOP	Ends the simulation.

TABLE II

LIST OF COMMANDS USED BY THE SIMULATOR.

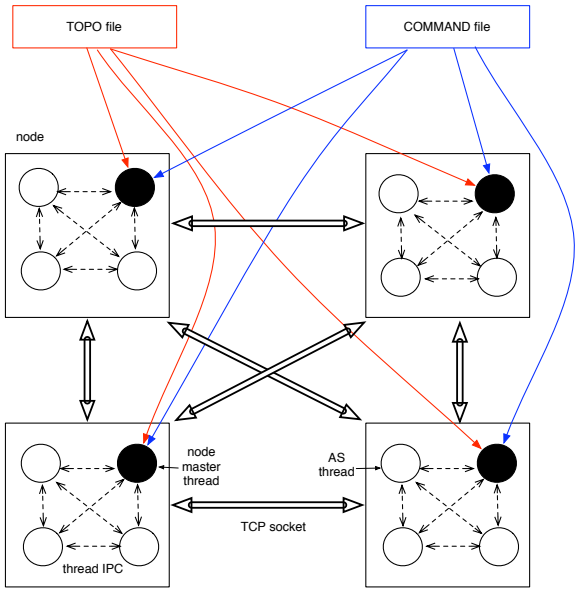

Fig. 1. Overview of the *lseb* simulator architecture. Threads communicate though IPC when on the same node, and over TCP sockets across nodes. The topology and command files control the simulation.

input to the simulator itself. Additionally, the processed data is used to generate a realistic Internet topology. We have developed the *bgptopo* utility that, for a user-specified set of dates, determines the neighbors of an AS based on BGP announcements and withdrawals as observed by one of the Route Views listening points. With this topology generator, we can reconstruct the state of the Internet at any given point in time.

The results of this processing are two files: a topology file contains information on ASes, the prefixes they encompass, and the links between them and other ASes, while a command file contains timing information to be used for determining when BGP updates, such as announcements and prefix withdrawals, should occur. Table I gives a list of commands used by the topology generator, while table II gives a list of commands used by the simulator.

The *lseb* simulator itself is written in Java. The operation of each individual AS is controlled by its own thread, and these are distributed across multiple computational nodes through assignment algorithms. Each simulation node has a master thread that dispatches commands to each AS thread. The simulation master thread dispatches commands to the node master threads. Communication is achieved between threads on the same node through thread IPC, while TCP sockets are used to transmit information between machines. Master threads communicate over priority channels that preempt any other communication. The masters on each node are contacted, and a wait cycle is executed by the simulation master thread after all nodes have been contacted. All of the discussed components are shown in figure 1.

Placing ASes on nodes is an open problem. Currently ASes are assigned either manually or in a round-robin fashion. To minimize a node's network communication load, we desire a heuristic that assigns neighboring ASes to the same simulation node, up to a node's IP size limit. Then most of the ASes assigned to a simulation node will form a connected graph, which minimizes the number of network messages that have to be exchanged between simulation nodes. We also want to ensure that nodes in a simulation testbed that are busy with other tasks get less ASes, or none. The DETER testbed is an ideal venue for this area of research, as the computational capacities and number of machines in the distributed cluster are known. We are developing a shim layer between DETER and *lseb* to indicate the activity levels of hosts. This will assist in developing node placement algorithms that maximize the available resources.

Figure 2 shows an example topology that was used during development of *lseb*. In this topology, there are 54 ASes distributed amongst four nodes in the system. In our simulated topology, there are multiple ASes administered per running process, with a master process acting as a coordinator for the slave processes. Each group of ASes is independently administered by their respective process, however, and computation is hence distributed across nodes running these processes. We also group all traffic sent between two simulation nodes within a time unit in a single network message to further reduce communication cost. We distribute routing information across simulation nodes so that each node only stores routing tables of the ASes it simulates. When traffic is generated, the simulation node determines the destination AS for the given IP address using shared data which maps IP ranges to ASes. The node

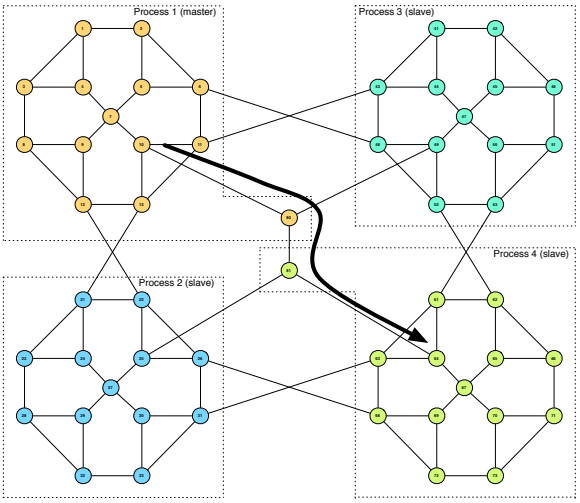

Fig. 2. The *lseb* simulator running a sample topology of 54 ASes across four nodes. The arrow indicates the best path as determined by BGP between AS 10 and AS 64.

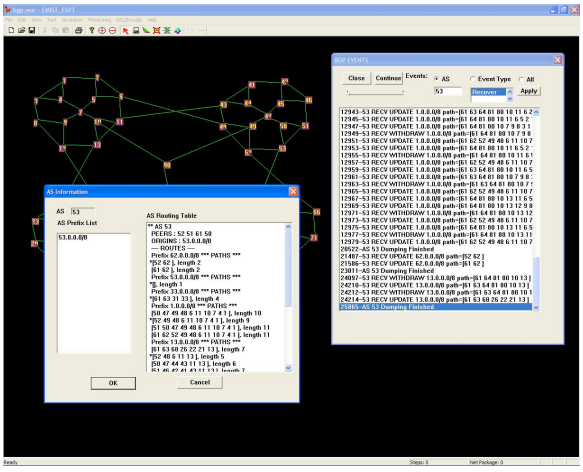

Fig. 3. The forensic visualization tool graphically displaying the sample topology and an associated event log for AS 53.

then uses its view of routing tables to calculate the path to the destination AS and the bandwidth consumption on this path. Some traffic may be dropped due to congestion. We calculate the portion of the traffic dropped and account for the bandwidth consumption and the drops at appropriate links but do not simulate the path of this traffic in the Internet. The rest of the traffic will either be delivered to another simulation node via a network message (if the AS path traverses more than one simulation node) or will generate a function call on the same simulation node.

IV. Visualization

We have developed a forensic visualization tool that allows for graphical representation of all ASes. An example of the visualization tool's output is displayed in figure 3. The tool takes topology data generated from the *bgptopo* extractor and represents connected links. It also spatially separates ASes by the computation nodes they are running on to provide an

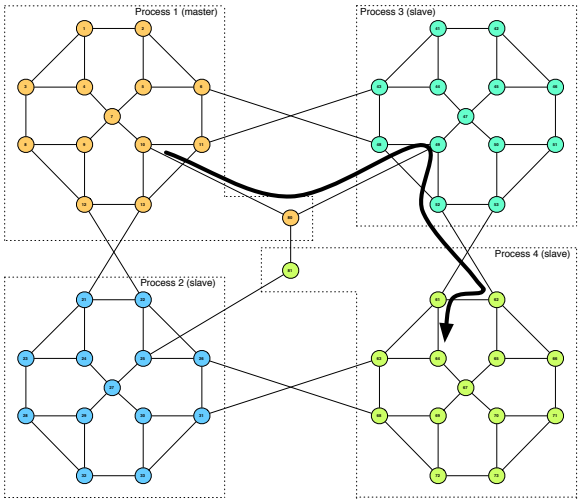

Fig. 4. The resulting network of 54 ASes and new path determined by BGP between AS 10 and AS 64 after link removal.

overview of the physical topology. Additionally, details such as event traces for specific ASes, and their associated routing tables, can be easily viewed.

V. Attack Simulation

BGP is responsible for routing information to its correct destination throughout the Internet. However, BGP is susceptible to many forms of attacks. Because it runs over TCP, sessions between BGP peers can be compromised by TCP attacks, such as resetting the session and causing a denial of service through a SYN flood. Attacks can also originate from the IP and physical layers, such as by link cutting, either through physical means or by congesting links between routers to block BGP and TCP heartbeat messages. If the adversary has the ability to cause links to oscillate by bringing connections up and down, they can force route dampening to occur; by manipulating the manner in which links come up and are brought down, it is possible to arbitrarily deny service to victim destinations indefinitely [19]. Figure 4 shows how *lseb* simulates link removal and how the BGP path selection algorithm chooses a different best path between ASes 10 and 64 in the example topology after the link cut.

Additionally, there are threats to routing that can be carried out through BGP. In particular, a misconfigured or malicious AS can advertise routing prefixes that do not belong to it, and claim that it originates these prefixes. This is called *prefix hijacking*. Because of the nature of routing in BGP, where shorter paths are generally preferred, the neighbors of a prefix advertising these falsely originated routes will be liable to believing them to be true. They will in turn start advertising these routes and because of the short path lengths, their neighbors in turn will start advertising these routes. This causes black holes to form around the areas where the hijacked prefix is advertised, denying any entities routing through this area from reaching the desired destination. These can be identified in current routing configurations as MOAS

conflicts, since multiple ASes will be advertising the prefix – the legitimate AS and the one hijacking the prefix. We have used *lseb* to simulate prefix hijacking by a rogue AS and the resulting change in BGP routing.

VI. Conclusions and Future Work

We have described our large scale simulator for BGP simulation, *lseb*. Using the DETER testbed, *lseb* is capable of simulating every AS in the Internet using realistic, data-driven topologies. We can replay network events and describe what-if scenarios using historical routing data. While the functionality is currently robust, further functionality is under development. We aim to add attack and defense modules so that different strategies for defending the greater Internet may be easily observed and modified. Additionally, we will compare the results of simulation to real organizational BGP data to determine how our coarse-grained approximations of routing compare to the actual routing process, which takes factors such as interior gateway protocols into account.

VII. Acknowledgments

We would like to thank Peng Liu and Lunquan Li for their work in developing the forensic visualization tools. Additional, we would like to thank Ihab Hamadeh for his help in adapting *lseb* to work on the DETER testbed.

References

[1] http://www.isi.edu/nsnam/ns/.
[2] http://www.ssfnet.org/.
[3] http://www.ece.gatech.edu/research/labs/MANIACS/GTNetS/.
[4] http://www-static.cc.gatech.edu/computing/compass/pdns/.
[5] W. Aiello, K. Butler, and P. McDaniel, "Path Authentication in Inter-domain Routing," Department of Computer Science and Engineering, Penn State University, Tech. Rep. TR NAS-TR-0002-2004, Network and Security Center, November 2004.
[6] W. Aiello, J. Ioannidis, and P. McDaniel, "Origin Authentication in Interdomain Routing," in *Proceedings of 10th ACM Conference on Computer and Communications Security (CCS)*. ACM, October 2003, pp. 165–178, washington, DC.
[7] A. Antony and H. Ujterwaal, "Routeing Information Service: R.I.S. Design Note," http://www.ripe.net/projects/ris/Notes/ripe-200.ps, Oct. 1999.
[8] G. Goodell, W. Aiello, T. Griffin, J. Ioannidis, P. McDaniel, and A. Rubin, "Working Around BGP: An Incremental Approach to Improving Security and Accuracy of Interdomain Routing," in *Proceedings of Network and Distributed Systems Security 2003 (NDSS)*. Internet Society, February 2003, pp. 75–85, san Diego, CA.
[9] Y.-C. Hu, A. Perrig, and M. Sirbu, "SPV: Secure Path Vector Routing for Securing BGP," in *ACM SIGCOMM*. ACM, August 2004.
[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, Apr. 2000.
[11] M. Liljenstam, D. Nicol, V. Berk, , and R. Gray, "Simulating Realistic Network Worm Traffic for Worm Warning System Design and Testing," in *Proceedings of the 2003 ACM Workshop on Rapid Malcode (WORM 2003)*, 2003.
[12] M. Liljenstam, Y. Yuan, and B. J. Premore, "A Mixed Abstraction Level Simulation Model of Large-scale Internet Worm Infestations," in *International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS 2002)*, 2002.
[13] D. Meyer, "The Route Views Project," August 2004, http://www.routeviews.org/.
[14] D. Moore, C. Shannon, G. M. Voelker, and S. Savage, "Internet quarantine: Requirements for containing self-propagating code," in *Proceedings of IEEE INFOCOM 2003*, San Francisco, CA, USA, July 2003.
[15] J. Ng, "Extensions to BGP to support secure origin BGP (soBGP)," Oct. 2002, Internet Draft.
[16] K. Perumalla and S. Sundaragopalan, "High-Fidelity Modeling of Computer Network Worms," in *Proceeedings of the Annual Computer Security Applications Conference (ACSAC)*, Tuscon, AZ, USA, Dec. 2004.
[17] S. Qiu, P. McDaniel, F. Monrose, and A. Rubin, "Characterizing address use structure and stabillity of origin advertizement in interdomain routing," in *11th IEEE Symposium on Computers and Communications*, Pula-Cagliari, Sardinia, Italy, June 2006.
[18] G. F. Riley, M. I. Sharif, and W. Lee, "Simulating Internet Worms," in *Proceedings of the 12th Annual Meeting of the IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS 2004)*, 2004.
[19] K. Zhang, S.-T. Teoh, S.-M. Tseng, C.-N. Chuah, K.-L. Ma, and F. Wu, "Performing BGP experiments on a semi-realistic internet environment," North American Network Operators Group (NANOG), October 2004.
[20] M. Zhao, S. W. Smith, and D. M. Nicol, "Aggregated path authentication for efficient BGP security," in *Proceedings of the 12th ACM Conference on Computer and Communications Security (CCS'05)*, Nov. 2005, alexandria, VA, USA.
[21] C. Zou, W. Gong, and D. Towsley, "Code Red Worm Propagation Modeling and Analysis," in *Proceedings of the 9th ACM conference on Computer and communications security*.