AutoNetkit: Simplifying Large Scale, Open-Source Network Experimentation

Simon Knight
University of Adelaide, Australia

Askar Jaboldinov
Loughborough University, United Kingdom

Olaf Maennel
Loughborough University, United Kingdom

Iain Phillips
Loughborough University, United Kingdom

Matthew Roughan
University of Adelaide, Australia

ABSTRACT

We present a methodology that brings simplicity to large and complex test labs by using abstraction. The networking community has appreciated the value of large scale test labs to explore complex network interactions, as seen in projects such as PlanetLab, GENI, DETER, Emulab, and SecSI. Virtualization has enabled the creation of many more such labs. However, one problem remains: it is time consuming, tedious and error prone to setup and configure large scale test networks. Separate devices need to be configured in a coordinated way, even in a virtual lab.

AutoNetkit, an open source tool, uses abstractions and defaults to achieve both configuration and deployment and create such large-scale virtual labs. This allows researchers and operators to explore new protocols, create complex models of networks and predict consequences of configuration changes. However, our abstractions could also allow the discussion of the broader configuration management problem. Abstractions that currently configure networks in a test lab can, in the future, be employed in configuration management tools for real networks.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations—*Network Management*

General Terms

Design, Experimentation, Management

1. INTRODUCTION

Emulated networks, which run a real router operating systems inside virtual machines, offer realistic yet inexpensive network experimentation. However they are time-consuming to configure, which limits their use in network research. AutoNetkit solves this problem, providing an integrated platform that allows the experimenter to visually describe their network experiment at a high-level, with resource allocation, device configuration handled automatically. The platform (shown in Figure 1) is Open-Source and generates router configurations from a network-level abstract representation, allocates appropriate resources, and deploys the network on emulated devices. This combination of simple specification with automated deployment and data collection allows for rapid iteration of network design leading to efficient evaluation of "what-if" scenarios, and providing a platform for research into complex policy and protocol interactions.

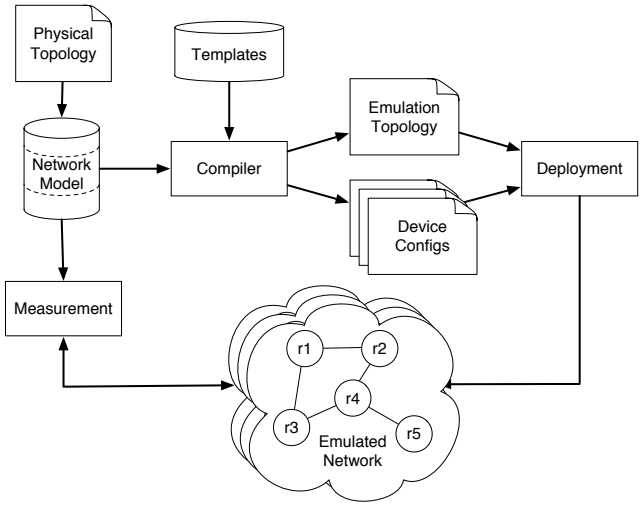

Figure 1: AutoNetkit: High-level descriptions are transformed into device-level configurations for deployment to an emulated network.

There are a number of platforms which can be used to conduct a network experiment on. Physical hardware networks are the most realistic, but are expensive, with a large scale testbed being beyond the budget of most researchers, and it is time consuming to manually configure and connect the physical routers. When available, hardware networks are typically small scale, limiting their usefulness in the study of large scale network behaviour.

Simulations, such as C-BGP [7], typically focus on one aspect of network behaviour, such as the routing decision process. This allows for large-scale experiments to be conducted on modest computer hardware, but at limited realism: modern networks are a complex interaction of multiple protocols, which cannot be easily simulated. Furthermore, interpretations of a standard (such as an RFC) can differ between vendor implementations, and bugs can exist in vendor software releases. To be relevant to real-world networks, the research community needs to study the behaviour of real vendor equipment, inclusive of RFC interpretations and bugs.

SIGCOMM'12, August 13–17, 2012, Helsinki, Finland.
ACM 978-1-4503-1419-0/12/08.

Emulated networks sit between these two extremes: by running the same operating system image as physical routers, they display the same protocol interactions and decision processes. Using virtual machines cuts costs dramatically with many virtual routers running on a commodity server.

However, emulated networks retain a main problem of hardware networks: the routers require configuration. For small networks, an experimenter may manually enter the configurations, but this approach doesn't scale well. While we may not need to plug cables into the router, the connection topology still needs to be created. In addition, IP addresses must be allocated and assigned to interfaces, routing protocols configured with appropriate AS numbers, IP addresses, and hostnames. This is time-consuming, tedious, and error-prone, and must be conducted before the actual network experiment itself is created. If the experiment is to be conducted on a different emulation platform (such as when comparing vendors) or if the network topology needs to be changed, the whole configuration process must be repeated.

2. AUTONETKIT

Emulation offers many advantages to network research. An automated configuration process will reduce the chance of errors, and free researchers to spend time on the experiments instead of setup. A standard platform reduces the space required in a paper to describe the experiment setup, and simplifies reproducibility of experiments. AutoNetkit addresses this configuration problem and simplifies the process of network experimentation.

The current version builds upon previous work [5], with a number of key new features. We allow the user to specify the network topology at a high-level, in a standard graph exchange format. This allows the use of a GUI-based graph editor, such as yED (`www.yworks.com/products/yed`) a freeware graph-drawing tool similar to Visio or OmniGraffle. In these graphs nodes correspond to routers, with edges indicating their connectivity. Node and edge attributes allow hostnames, AS numbers, and link weights to be specified. From these IGP and BGP topologies can be inferred (e.g. an eBGP session is created for a link that spans two Autonomous Systems). Additional attributes allow for advanced iBGP topologies and DNS databases to be configured. Autonetkit supports both OSPF and IS-IS IGPs, iBGP design patterns, and a clean abstraction for inter-domain BGP policy.

AutoNetkit supports Netkit [6], and can create and deploy to Junosphere [2], Dynamips [1], and C-BGP, and provides a tool to run measurement commands (such as traceroute or viewing the routing table) across the emulated network. It is easy to deploy the same experiment to multiple platforms: only the emulation server needs to be changed.

3. PERFORMANCE

As an example, we used AutoNetkit to setup a *bad gadget* [3]. We were able to setup and deploy this case-study in less than ten minutes to each of our supported emulated platforms. We verfied the expected oscillatory behaviour through automated traceroutes. On Cisco IOS and Juniper Junos routers the IGP tie-break worked as expected. However, the routing did not oscillate on Quagga and C-BGP — due to a difference in one of the final steps of the BGP best-path selection process. This example illustrates both the importance of testing behaviour on real router platforms, and the ease of doing so with AutoNetkit.

While AutoNetkit allows rapid design and testing of small case studies, it is especially useful for large networks. Our combination of simple visual network design and multi-AS support complete with BGP policy allows for realistic large-scale network experiments.

In another example, we have created a 89-node 132-link network topology, based on example networks from The Internet Topology Zoo [4]. AutoNetkit generated the configuration files for this network in less than 3 seconds on a 2007 MacBook Pro. The time taken to deploy these configuration files to a number of emulation platforms are shown in Table 1.

Platform	Memory (GB)	Startup Time (sec)
Netkit	12.6	402
Dynagen/GNS3	20.9	55
Junosphere	136.8	1011
C-BGP	0.020	< 0.2

Table 1: Resource use for 89 router example network.

4. SUMMARY

Our work provides an environment for network researchers to play and experiment. The tool takes a high-level abstract representation of a network and compiles it to a choice of emulated environments. The result is a level of scalability and flexibility that is hard to achieve in a hardware testbed, but with substantially more realism than a simulation.

AutoNetkit is fast and is capable of building network configurations in seconds, even for large networks. By using a diagram-based, high-level abstraction we allow a fast capture of the network design, enabling the examination and investigation of complex "what-if" scenarios.

AutoNetkit is currently used by industry, in teaching, and the networking research community. It is open-source and written in Python, providing a multi-platform tool that is freely available online at `www.autonetkit.org`.

5. ACKNOWLEDGMENTS

This project was supported by the Australian Government through an Australian Postgraduate Award, Australian Research Council Discovery Grants DP110103505 and DP0985063, and Cisco Grant 2011-89493(3696). We also would like to thank Hung Nguyen, Nick Falkner and Michael Rumsewicz for their helpful input.

6. REFERENCES

[1] Dynamips: Cisco 7200 simulator.

[2] Junosphere. `http://www.juniper.net/as/en/products-services/software/junos-platform/junosphere/`.

[3] T. Griffin, F. B. Shepherd, and G. Wilfong. The stable paths problem and interdomain routing. *IEEE/ACM Trans. on Networking*, 2002.

[4] S. Knight, H. Nguyen, N. Falkner, R. Bowden, and M. Roughan. The Internet Topology Zoo. *Selected Areas in Communications, IEEE Journal on*, 29(9):1765–1775, 2011.

[5] 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.

[6] M. Pizzonia and M. Rimondini. Netkit: easy emulation of complex networks on inexpensive hardware. In *TridentCom*, 2008.

[7] B. Quoitin. C-BGP, an efficient BGP simulator. http://cbgp.info.ucl.ac.be/, 2003.