Panopticon: Supervising Network Testbed Resources

Marc Werner, Daniel Kratschmann, Matthias Hollick
Technische Universität Darmstadt, Secure Mobile Networking Lab (SEEMOO)
{mwerner, dkratschmann, mhollick}@seemoo.tu-darmstadt.de

***Abstract*—Wireless communication technology is pervasive and forms a large portion of our communication infrastructure. However, large scale testbeds for the evaluation of new and upcoming systems are required to benchmark newly developed mechanisms in a real world environment. While a plethora of wireless testbeds already exist, they are often highly underutilized as current testbed management solutions only allow one experiment to run at a time, and experiments often require only a small number of nodes. This is especially true during the development of new mechanisms where a small setup is sufficient to test the general functionality. In this demonstration we present Panopticon, a large-scale testbed management solution that allows for partitioning wireless testbeds into small slices where each slice can run its own experiments without interfering with other currently running evaluations. We use dynamic network partitioning by utilizing Virtual Local Area Networks (VLANs), creating self-contained environments for existing frameworks to run in. The system is implemented as a Service Oriented Architecture (SOA) and provides a run-time environment for highly specialized experiment frameworks that target the exact needs of the currently running evaluation. Panopticon is thus able to increase the utilization of testbed environments by partitioning the testbed into multiple slices while still giving experimenters exclusive access to the managed resources.**

I. Introduction

Research in the area of wireless networks requires the use of testbeds to investigate the behavior of systems and protocols under real world conditions. The full potential of newly implemented protocols and services can only be explored in large-scale scenarios where dozens to hundreds of nodes communicate in one single network.

However, during development of such protocols and services it is often sufficient to only access a few devices to rapidly deploy and test the implementation.

Existing specialized testbed frameworks such as OMF [1], WISEBED [2] and Emulab [3] typically require the exclusive allocation of resources to a single experimentation framework. Additionally, the existing frameworks often do not allow for the simultaneous use of the testbed by multiple researchers but assign the complete infrastructure to a single experiment. Some frameworks circumvent this limitation by virtualizing the testbed hardware, thus sacrificing unlimited access to the underlying components.

Our demonstration shows a possible solution to this dilemma. We propose a scalable and modular testbed management system called Panopticon that uses Virtual Local Area Networks (VLANs) in conjunction with additional services to partition the testbed. These partitions serve each experimenter with the desired environment and allows for direct and unrestricted access to the allocated resources.

In the remainder of this paper, we introduce Panopticon and propose a demonstration setup to show its potential for testbed management.

II. Panopticon

Panopticon is designed as a Service Oriented Architecture (SOA) and its functionality is composed from multiple services that are each responsible for a distinct feature. An overview of the architecture is depicted in Fig. 1.

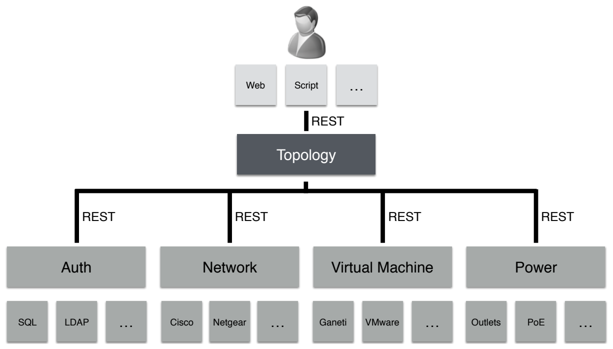


Fig. 1. Overview of the Panopticon architecture.

All services are based on a common framework that offers basic functionality shared across all loaded services. This ranges from database connectivity and abstraction using a Object Relation Mapper (ORM), over a caching backend utilizing a Redis [4] store, to a common Standard Data Library (STDL) that allows us to specify resources and network topology.

The framework provides methods to communicate with networked devices such as switches using the Simple Network Management Protocol (SNMP). The implementation abstracts from vendor specific SNMP implementations and thus allows Panopticon to access different networking hardware in an agnostic way. The abstraction is done by introducing an injection engine that transforms the internal representation of a configuration command into a device specific sequence of SNMP commands specified by a Domain Specific Language (DSL). The injection engine also chains multiple SNMP commands if possible thus speeding up execution on the target device, and provides entry points to support devices with non-standard behavior.

Additionally, the Panopticon service framework includes mechanisms to authenticate users and services against the central Authentication Service (AS). The authentication engine offers a three-tier model for authentication and authorization using permissions, privileges and mandates. While the first two allow access to the API and the testbed resources, the latter is used to delegate privileges to another user or service. A service is thus able to execute commands on behalf of an authenticated user and access restrictions for this user also apply to the service.

A. *Services*

All services are based on the common Panopticon framework and are able to communicate with other services in the network using Representational State Transfer (REST) interfaces. Data transfered between services is encoded using the JavaScript Object Notation (JSON) enabling effortless interoperability with external services and frameworks such as the libraries used in the web frontend.

1) Network Service: The Network Service (NWS) controls the management network of the testbed. It is responsible for assigning experiment VLANs to switch ports and manages the central border router that connects the experiment network to external networks such as the Internet.

The NWS has a global view of the complete physical network, the connected testbed nodes as well as the virtual machine hosts. It runs a central optimization algorithm that finds the forwarding path between all resources assigned to an experiment and implements the necessary settings on all ports along this path.

2) Authentication Service: The AS is responsible for handling authentication and authorization of users as well as services using mandates. Other services use the authentication engine described above to connect to and interact with the AS. The service connects to a directory server using the Lightweight Directory Access Protocol (LDAP). The AS retrieves the information stored in the directory and maps the information to Panopticon resources using a predefined ruleset. These mappings enable a user function level access.

3) Virtual Machine Service: The Virtual Machine Service (VMS) provides a unified interface to various hypervisors such as Ganeti [5]. The service manages the complete lifecycle of virtual machines in a partition.

It is responsible for provisioning virtual machines with a basic installation of an operating system as well as a network configuration and additional software required to manage the assigned testbed partition.

Researchers can use the virtual machines to run existing testbed management systems on top of Panopticon and utilize these systems to control the experiments in the assigned network slice.

4) Power Service: The Power Service (PS) is responsible for controlling the supply of electric power to the testbed nodes either by managing Power over Ethernet (PoE) ports on a network switch or switchable sockets integrated into the node's hardware.

5) Topology Service: The Topology Service (TS) orchestrates Panopticon's services to form a complete testbed management system. It provides the concept of experiments to the user. An experiment combines resources from other services to a unified view of the testbed network where resources can be dynamically added or removed depending on the needs of the user as well as the current time.

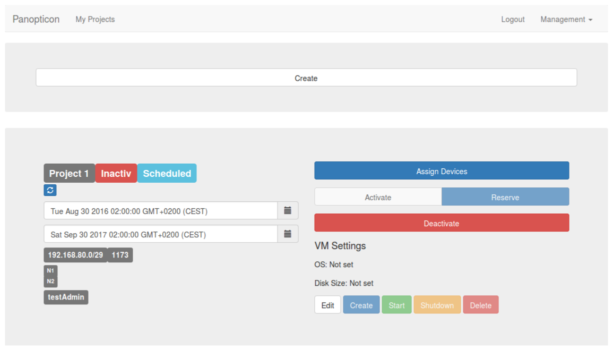
The TS integrates a scheduler that handles time dependent resource allocation. It is able to dispatch experiments in the future by reserving and automatically allocating the resources needed once the scheduled start date has arrived as well as deallocate the resources from experiments that have reached their expiration date. The scheduler can also propose alternate experimentation time slots if the required nodes are in use during the initially requested slot.

Additionally, the TS is responsible for allocating IP subnets to each experiment and reclaiming the addresses once the experiment has expired. This centralized allocation of addresses is necessary to enable routing outside the experiment network and allow access to external networks for running experiments.

The TS is able to communicate with multiple backend services of the same type, such as multiple NWS or PS, and thus control a testbed that is distributed over a large area and potentially under different administrative control. The placement of services locally close to the actual hardware is advisable as this reduces the RTT between the service and the controlled devices.

B. *Frontend*

Panopticon offers a web based frontend to interact with the TS as shown in Fig. 2. It allows users to create, view and manage experiments and associated resources.



Tue Aug 30 2016 02:00:00 GMT+0200 (CEST)

Sat Sep 30 2017 02:00:00 GMT+0200 (CEST)

192.168.80.0/29 1173

N1

N2

testAdmin

VM Settings

OS: Not set

Disk Size: Not set

Fig. 2. Screenshot of the Panopticon frontend.

The installation of upper layer management solutions is supported through the frontend by selecting an appropriate virtual machine during experiment creation, and the frontend allows the user to directly interact with the associated hardware during the execution of an experiment.

Additionally, the frontend provides an interface to the scheduler of the TS to appoint a time for experiment runs on the network.

III. System Demonstration

We use a small example network (see Fig. 3) to show the capabilities of Panopticon during the demonstration. While Panopticon was specifically designed with large scale testbeds in mind, it also shows its potential when only a small set of nodes need to be configured.

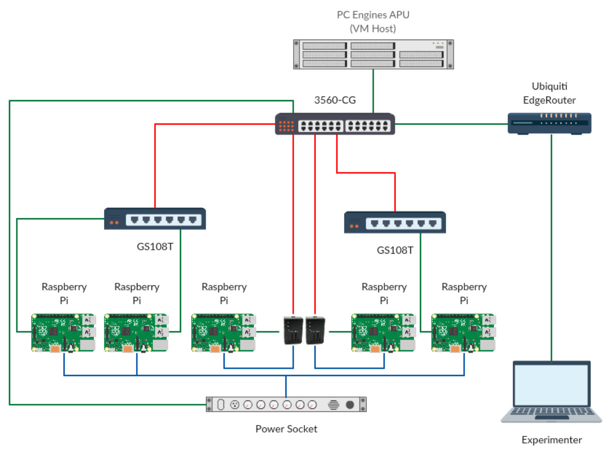


Fig. 3. Network setup during the demonstration.

We present the basic operation of the system as well as the unique characteristics of our testbed management system. Specifically, we show:

1) The creation of an experiment including the search and filter functions for testbed nodes as well as their association to the experiment.
2) The creation and management of virtual machines when setting up and running an experiment.
3) The control of electric power supplies during an experiment to turn the nodes on and off.
4) The concurrent execution of multiple experiments where each run is assigned a subset of nodes.
5) The scheduling of experiments if the testbed is currently fully utilized. This also includes the automatic allocation of an appropriate timeslot using the scheduler of the TS.

Furthermore, we show how a seamless handover between experiments is possible when running Panopticon as the management system, thus leading to a higher utilization of the available testbed resources.

IV. Demonstration Requirements

We will supply the following equipment to demonstrate the features of Panopticon:

- Five Raspberry Pi Model 3 (RPi) that serve as testbed nodes. The hardware is equipped with USB Wi-Fi dongles that allow wireless connectivity among the RPi 3 and can be freely configured by the experimenter. Two RPis will be powered via PoE using a separate splitter while the other 3 devices are powered via USB power adapters connected to a switchable socket.
- Two eight port Netgear GS108T manageable switches functioning as leaf switches that connect the wireless nodes to the testbed management network. The devices are powered over PoE and thus do not require a separate power socket.
- One ten port Cisco Catalyst 3560-CG PoE enabled switch that serves as an aggregate switch for the management network of the testbed as well as a power source to the PoE powered RPis and the leaf switches.
- One Ubiquiti EdgeRouter Lite that serves as a gateway between the testbed management network controlled by Panopticon and any external networks.
- One ANEL NET-PwrCtrl HOME switchable power socket with three switchable outlets and one fixed powered outlet.
- One PC Engines APU 1d4 computer with an AMD T40E, 1 GHz dual core CPU, 4 GB of memory and 30 GB of SSD storage running the Ganeti 2.12 virtual machine manager on top of Debian Jessie. The system serves as a virtual machine host that provides VMs to the configured experiments that run the actual testbed software.
- One laptop computer to show the GUI of Panopticon and to allow the audience to dynamically configure our demonstration network.

The demonstration requires 1.5 meters of bench space to setup the switches, router and testbed nodes as shown in Fig. 4.

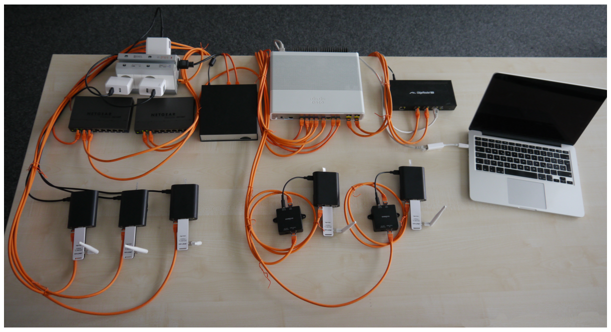


Fig. 4. Setup of the demonstration.

Additionally, the setup requires a power supply with six outlets to power the basic components while the peripherals are either powered via PoE or via switchable power sockets provided by us. The setup is self-contained and does not require an Internet connection.

V. Summary

We described Panopticon, a testbed management solution that allows partitioning of large scale testbeds. It increases the utilization of testbeds by assigning slices of testbed resources to multiple concurrent experiments while still providing the experimenter with direct access to the hardware.

Acknowledgment

This work has been funded by the DFG as part of the CRC 1053 MAKI and the LOEWE Priority Program NICER.

References

[1] T. Rakotoarivelo *et al.*, “OMF: A Control and Management Framework for Networking Testbeds,” *ACM SIGOPS OSR*, Jan. 2010.
[2] G. Coulson *et al.*, “Flexible Experimentation in Wireless Sensor Networks,” *Commun. ACM*, Jan. 2012.
[3] B. White *et al.*, “An Integrated Experimental Environment for Distributed Systems and Networks,” in *OSDI*, 2002.
[4] Sanfilippo, Salvatore and Noordhuis, Pieter, “Redis: An open source, in-memory data structure store,” http://www.redis.io.
[5] Google, Inc., “Ganeti virtual machine cluster management,” http://www.ganeti.org.