An experimental platform for home wireless router

Xuefeng Huang

***Abstract*—As one of the most economically significant and fastest growing sectors of the Internet, broadband networks have attracted interest from researchers. To better understand broadband networks, we developed Seattle, an open research and educational testbed that utilizes computational resources provided by end users on their home wireless routers with custom firmware (OpenWrt [2]). Unlike most other platforms, Seattle provides a privacy protection of embedded device data and maintains the security of donated device from potentially buggy experiment codes. We find that our platform is flexible enough to implement a variety of network measurements despite its security restrictions. This paper discusses some of the challenges we faced building and using a platform for deploying measurement in home networks, describe its design and implementation.**

I. Introduction

Currently it has been difficult to study home networks on a large scale because network technologies like network address translators (NATs) present only an opaque view of the home network to the global Internet. To better understand home networks, an experimental platform should be hosted in home networks, to provide visibility into the missing part of Internet. While previous works [9] have studied access networks from home gateway, they are unable to let researchers run arbitrary codes without compromising the privacy of the user and abuse of host or network resources. We present Seattle, a platform for measurement experimentation from home gateway. Seattle is able to deploy in home wireless router so that it is capable of running both active and passive experiments from a vantage point between the access ISP and the home network, as shown in Fig. 1.

In this paper, we introduce Seattle Testbed [6], a distributed cloud platform that allows researchers to run their project on system worldwide. This testbed provides secure data access while preserving user privacy. Through a programmable interface on device, the testbed enables researchers to deploy a wide range of network measurements. We also discuss the constraints we faced in the design, implementation and deployment of Seattle.

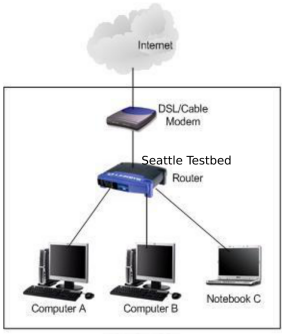

Fig. 1. The router sits directly behind the modem in the home network.

II. Challenges and Contributions

A. Challenges

A testbed that allows researchers to execute code across donated embedded devices faces four major challenges:

- First, the embedded devices poses a limited resources challenge. It is hard to run heavy scripting languages like Python or Ruby.
- The second challenge is user's networking experience. Seattle nodes are on the direct path of real Internet users. A malicious network measurement experiment will noticeably affect a normal user's network experience.
- The third challenge is secure data access. Network traffic data pose a risk to device donors

whose devices are exposed to malicious code written by users.

- The fourth challenge is data privacy. Users should not gain access to information that donors don't want to share.

As the development of hardware technology and based on our practical experience, running Python codes are going well on embedded devices. And the last three challenges can be solved via using a secure and performance-isolated sandbox and a reference monitor framework we proposed.

B. Contributions

1) Home routers platform: This platform is based on Seattle [10], a community-driven, open-source cloud computing system. Compared to computer and mobile device environments, deployment on home wireless router has more resource limitation such as restricted computational resources. However, recently we are able to port Seattle to embedded devices. Users can build their own Seattle installer (IPK) via config file we provide using OpenWrt SDK and install it on the device directly. Currently, I deploy this platform in TP-Link TL-WDR3600 router, which has a 560 MHz MIPS processor, 8MB of flash storage, 128MB of RAM and a dual-band wireless interface.

2) Extensions to Seattle Testbed: Our testbed implements new research capabilities for home wireless router by improving on the Seattle sandbox. To handle home wireless routers, our testbed uses low-level system calls in the OpenWrt platform [2] with the Restriction Python (Repy) [7], the core sandbox of Seattle. In order to securely interact with home routers on remote user devices, we use Fence [8] to allocate a fixed percentage of the device's CPU, memory disk, and other resources to one or more VMs. For example, we set the legal times of accessing proc file system to prevent Dos attack using our API calls. Our testbed adds seven functionalities based on Repy, as shown in in Table 1. Through these new functionalities I proposed, researchers are able to implement a wide range of network measurements, as shown in in Table 2.

API	Description
scan	Collect the list of access points found with a WiFi scan. For each access point we collect BSSID, SSID, signal strength and channel number.
get_station	Record downlink statistics per associated client (e.g., Total packets sent, received, retried, client's signal strength at home wireless router).
get_network_interface	Return a list of available network interfaces.
get_network_bytes	Record information about the configured network interfaces. The statistics include metrics such total number of received or transmitted bytes, drops, errors.
get_network_packets	Record information about the configured network interfaces. The statistics include metrics such total number of received or transmitted packets, drops, errors.
ping	A pure python ping implementation using raw sockets.
traceroute	Return the route packets take to network host.

TABLE I
NEW API

Type	Parameters	Descriptions
Passive	Aggregate traffic statistics per associated client (e.g., Total packets sent, received, retried, clientś signal strength at AP) neighboring APs information	Home network characteristics Usage characteristics
Active	Throughput, Latency, Loss, Jitter, traceroute, DNS lookups	ISP characteristics Internet connectivity and reachability

TABLE II
EXPERIMENTS OBTAINED FROM SEATTLE

3) Experimental Characterization of home wireless networks: By integrating Seattle with the home network, we get the benefits of a real world deployment while ensuring flexibility to run experiments without compromising home network. We demonstrate Seattle's utility by implementing one example usage case that together exercise different new API calls: monitoring how busy the WiFi channels in a place over one or two weeks. Our goal is to illustrate how Seattle can support disparate measurement needs.

III. STATUS AND FUTURE WORK

We have currently deployed Seattle in a router in the NYU lab. This device is added to Poly VlAN 147. Instead of traditional installer, Seattle is able to be installed via OPKG package manager [3] [4]. It is better to resolve dependencies with packages in the repositories. In order to let Seattle group members test these new capabilities in home wireless router easily, I provided documentation and

some example programs [5]. Currently. those new API calls work fine form group members' feedback. I am currently studying the network traffic of home networks. Our plan is to analyze the periodicity of usage patterns in various home networks. The data we collect will include aggregated traffic information and timestamp. I have ran an experiment to collect data using Seattle in the home wireless router. However, due to the issue [1] about GETTID doesn't work in OpenWrt, I only can get aggregated traffic information.

In few weeks, I will fix unsolved issue first and then demonstrate its utility in one study: monitoring how busy the WiFi channels in a place over one or two weeks. This study can give researchers a good example to help them to write and launch their own measurements. I am currently writing my master thesis draft, it includes five sections: introduction, challenges and contributions, related work, a secure network testbed design, implementation. After obtaining actual experiment result, I will start working on evaluation and conclusion section.

REFERENCES

[1] Linux 'gettid' syscall problem on openwrt. https://github.com/SeattleTestbed/repy_v2/issues/98.

[2] Openwrt. https://openwrt.org/.

[3] Opkg package manager. https://wiki.openwrt.org/doc/techref/opkg.

[4] Seattle-openwrt. https://github.com/XuefengHuang/seattle-package/tree/newapi.

[5] Seattle testing introduction. https://github.com/XuefengHuang/seash/tree/test.

[6] Justin Cappos, Ivan Beschastnikh, Arvind Krishnamurthy, and Tom Anderson. Seattle: a platform for educational cloud computing. In *ACM SIGCSE Bulletin*, volume 41, pages 111–115. ACM, 2009.

[7] Justin Cappos, Armon Dadgar, Jeff Rasley, Justin Samuel, Ivan Beschastnikh, Cosmin Barsan, Arvind Krishnamurthy, and Thomas Anderson. Retaining sandbox containment despite bugs in privileged memory-safe code. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 212–223. ACM, 2010.

[8] Tao Li, Albert Rafetseder, Rodrigo Fonseca, and Justin Cappos. Fence: protecting device availability with uniform resource control. In *Proc. of USENIX ATC*, 2015.

[9] Srikanth Sundaresan, Sam Burnett, Nick Feamster, and Walter de Donato. Bismark: A testbed for deploying measurements and applications in broadband access networks. In *2014 USENIX Annual Technical Conference (USENIX ATC 14)*, pages 383–394, Philadelphia, PA, June 2014. USENIX Association.

[10] Yanyan Zhuang, Albert Rafetseder, and Justin Cappos. Experience with seattle: A community platform for research and education. In *Research and Educational Experiment Workshop (GREE), 2013 Second GENI*, pages 37–44. IEEE, 2013.