SRFuzzer: An Automatic Fuzzing Framework for Physical SOHO Router Devices to Discover Multi-Type Vulnerabilities

中国科学院信息工程研究所
INSTITUTE OF INFORMATION ENGINEERING,CAS

{**Yu Zhang**, Wei Huo, Kunpeng Jian, Ji Shi, Haoliang Lu, Longquan Liu, Chen Wang, Dandan Sun}[1,2], Chao Zhang[3], Baoxu Liu[1,2]

Institute of Information Engineering, Chinese Academy of Sciences[1]

School of Cyber Security, University of Chinese Academy of Sciences[2]

Institute for Network Science and Cyberspace, Tsinghua University[3]

Outline

Introduction

- Security of the SOHO Router is Important
- How to Find the Vulnerability
- Challenge of Fuzzing the SOHO Router

Example—— NTP configuration

SRFuzzer——Our Solution

- Seed Generation
- Seed Mutation
- Exceptional Behavior Triggering and Monitoring
- Power Control

Evaluation

- Experiment Overview
- Analysis of Issues
- Performance of Monitors
- Comparison with Popular Fuzzers

Discussion

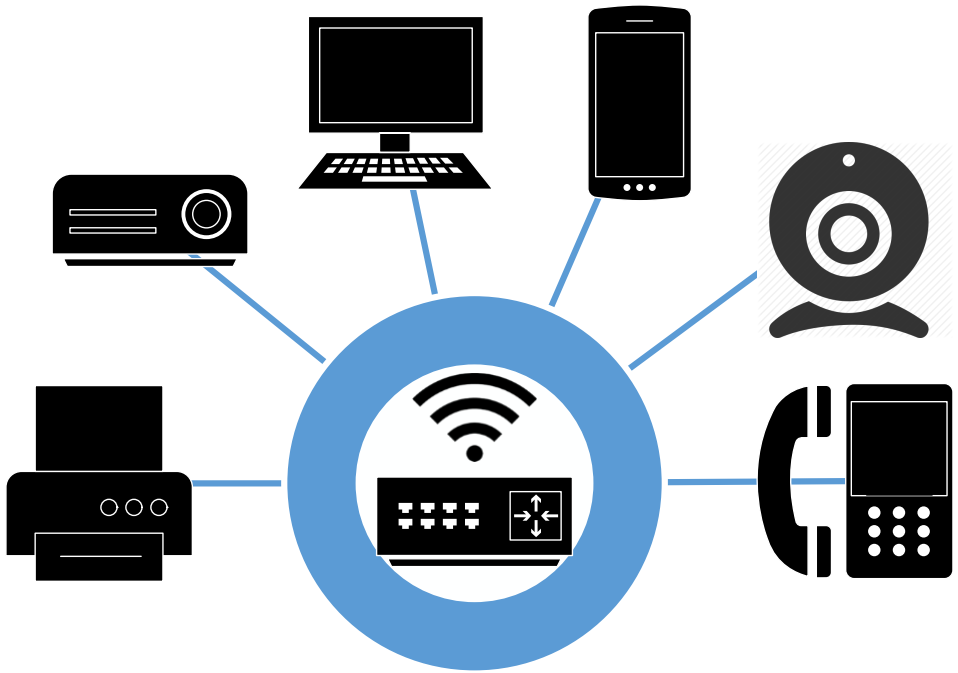
Summary

Q&A

Security of the SOHO Router is Important

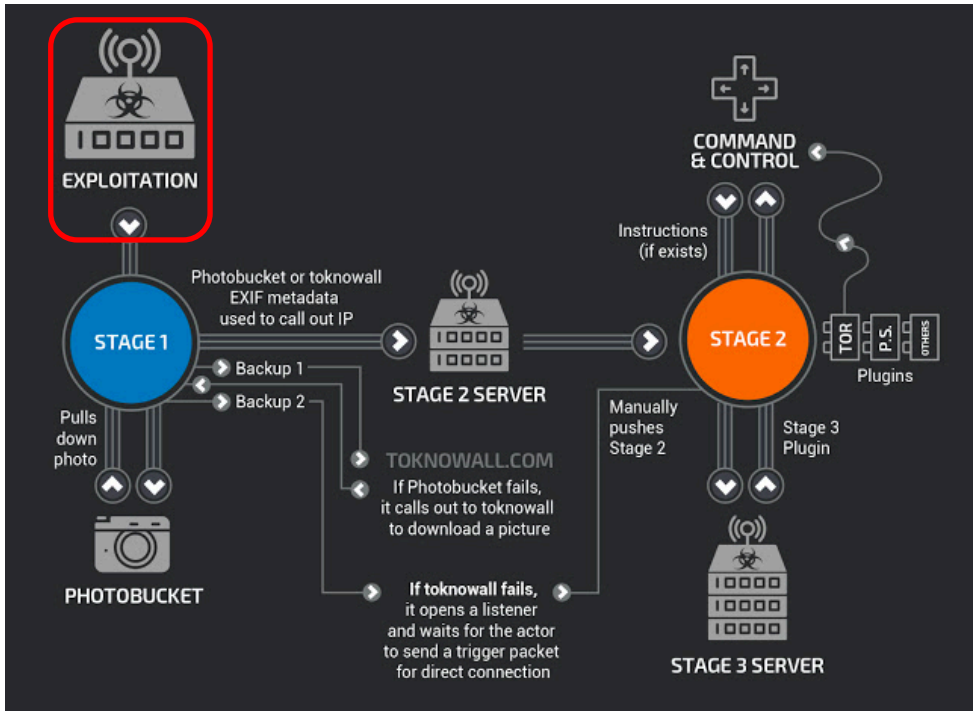
SOHO routers are in prominent position in nowadays life.

- Smart phone, personal computer, camera, printer, etc



SOHO routers are one of the essential exploiting targets by adversaries.

- VPNFilter infected at least 500,000 devices in at least 54 countries [1]

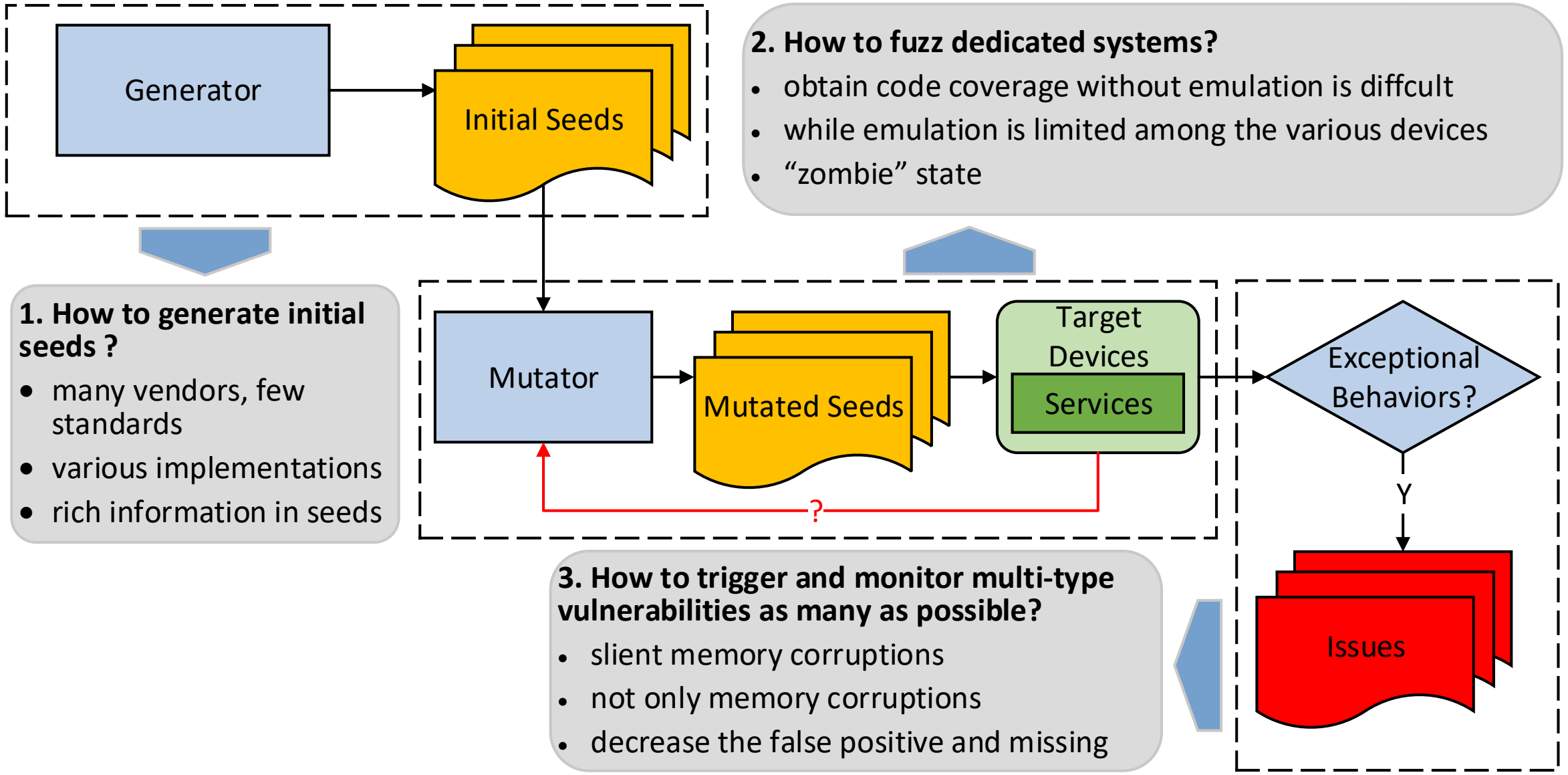


Find Vulnerabilities in Routers

Fuzzing is popular in discovering vulnerabilities of IoT devices

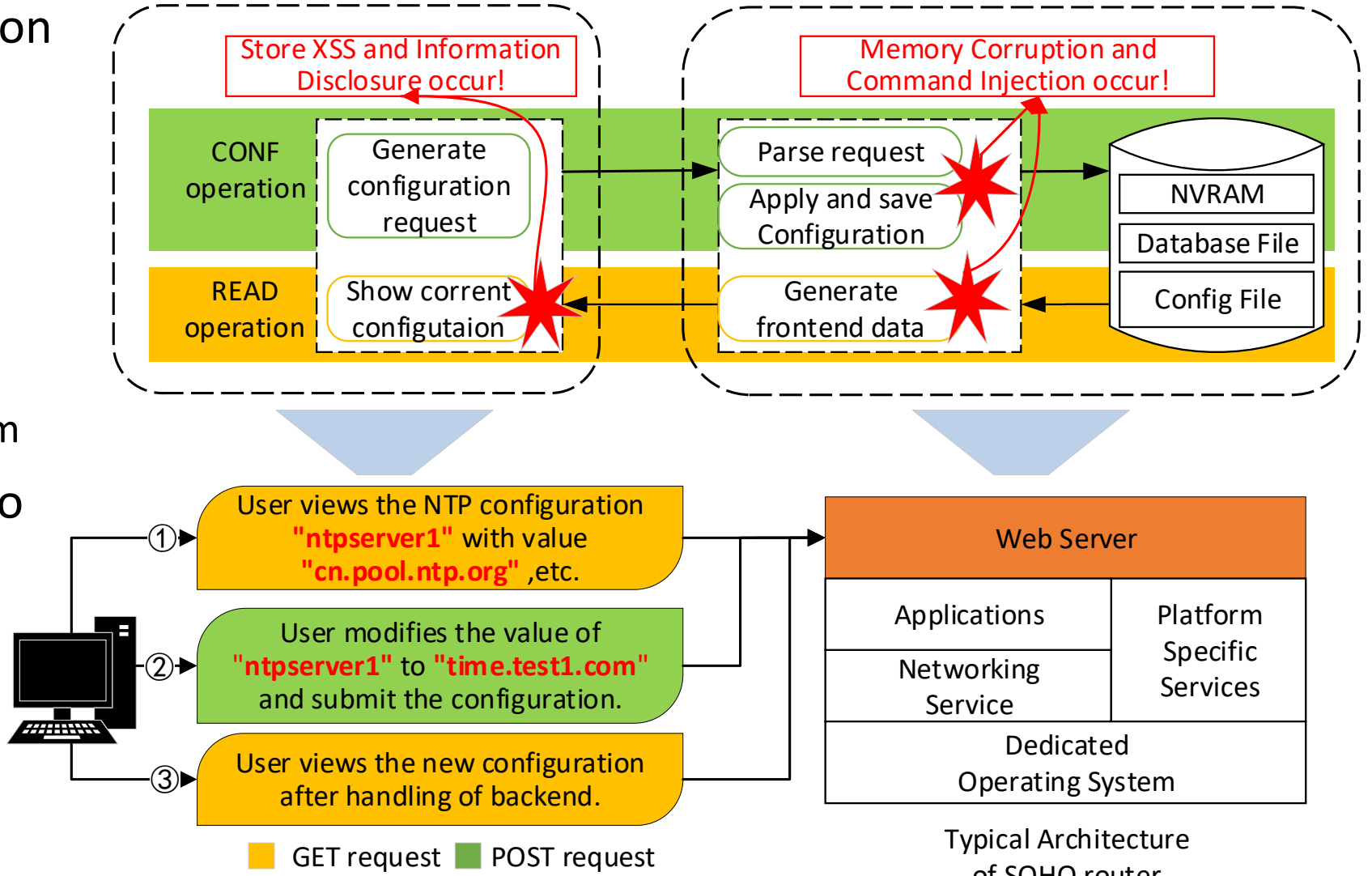
Technique	Andrei Costin et al.	FIRMADYNE	IoTFuzzer	Muench et al.	FIRM-AFL
Technique	Static and Dynamic Analysis	PoC	Blackbox Fuzzing	Blackbox Fuzzing	Greybox Fuzzing
Emulation	YES	YES	NO	YES	YES
Multi-Type Vulnerability	YES	NO	NO	NO	NO
Zero-Day Detection	YES	NO	YES	NO	YES
Coverage-guide	NO	NO	NO	NO	YES

Challenge of Fuzzing the SOHO Router



Example——NTP configuration

- CONF-READ communication model
  - GET request is a READ operation
  - POST request is a CONF operation
- KEY-VALUE data model
  - ntpserver1=time.test1.com
- Several different phases to trigger multi-type vulnerabilities



Example——NTP configuration

- 2 functions to handle the variable ntpserver1
- A command injection vulnerability in conf_ntpserver1() function
  - ➢ Data type inconsistency
- A stack-based overflow vulnerability in read_ntpserver1() function
  - ➢ Length limitation inconsistency in 2 related functions
- The memory corruption can cause crash, what about the command injection, XSS and info disclosure?

Raw Request

```
POST /apply.cgi?/NTP_debug.htm HTTP/1.1
Host: 192.168.66.1
Connection: keep-alive
Content-Length: 209
submit_flag=ntp_debug&conflict_wanlan=&ntpserver1=time.test1.com
&ntpserver2=time.test2.com&ntpadjust=0&hidden_ntpserver=GMT8&h
idden_dstflag=0&hidden_select=33&dif_timezone=0&time_zone=GMT-
8&ntp_type=0&pri_ntp=
```

ntpserver1=;reboot;

```
1 int conf_ntpserver1(char * input){
2   char buf[0x100];
3   char * ntp = read_from_request("ntpserver1", input);
4   if(strlen(ntp) > 0x80)
5     return -1;
    //use varaiable "ntp" to build config command.
6   sprintf(buf, "/usr/bin/config ntpserver=%s.", ntp);
    //command injection occurs.
7   system(buf);
8   return 0;
}
```

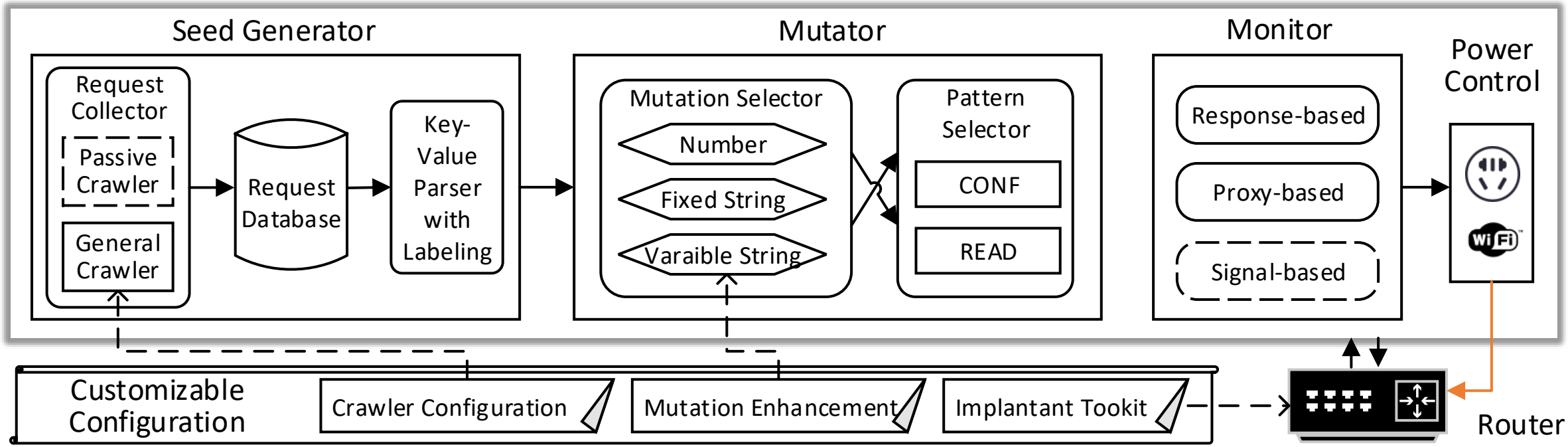
ntpserver1=aaa……aaa

0x70

```
9 int read_ntpserver1(){
    //the length of info is no more than 0x80.
10  char info[0x50];
11  char * ntp = get_config("ntpserver1");
    //stack-based overflow occurs.
12  sprintf(info, "ntpserver=%s", ntp);
13  return 0;
}
```

SRFuzzer

- Fuzz the physical devices directly and automatically
- Trigger multi-type vulnerabilities with KEY-VALUE data model and CONF-READ communication model
- Generate information and monitor it when triggering exceptional behaviors
- Use smart plug to restore the device from “zombie” state
- Modular design and well extendibility



Seed Generation

- Crawler
  - General crawler
  - Passive crawler
- KEY-VALUE parser with labeling
  - Variable string
  - Fixed string
  - Number

Raw Request

```
POST /apply.cgi?/NTP_debug.htm HTTP/1.1
Host: 192.168.66.1
Connection: keep-alive
Content-Length: 209
submit_flag=ntp_debug&conflict_wanlan=&ntpserver1=time.test1.com
&ntpserver2=time.test2.com&ntpadjust=0&hidden_ntpserver=GMT8&h
idden_dstflag=0&hidden_select=33&dif_timezone=0&time_zone=GMT-
8&ntp_type=0&pri_ntp=
```

Seed

URL: http://FUZZING_IP/apply.cgi?/NTP_debug.htm
METHOD: POST
Tuple SET:

key	value	attributes
submit_flag	ntp_debug	fixed str, variable str
conflict_wanlan		variable str
ntpserver1	**time.test1.com**	**variable str**
hidden_dstflag	0	number, variable str
hidden_select	33	number, variable str
...	...	...

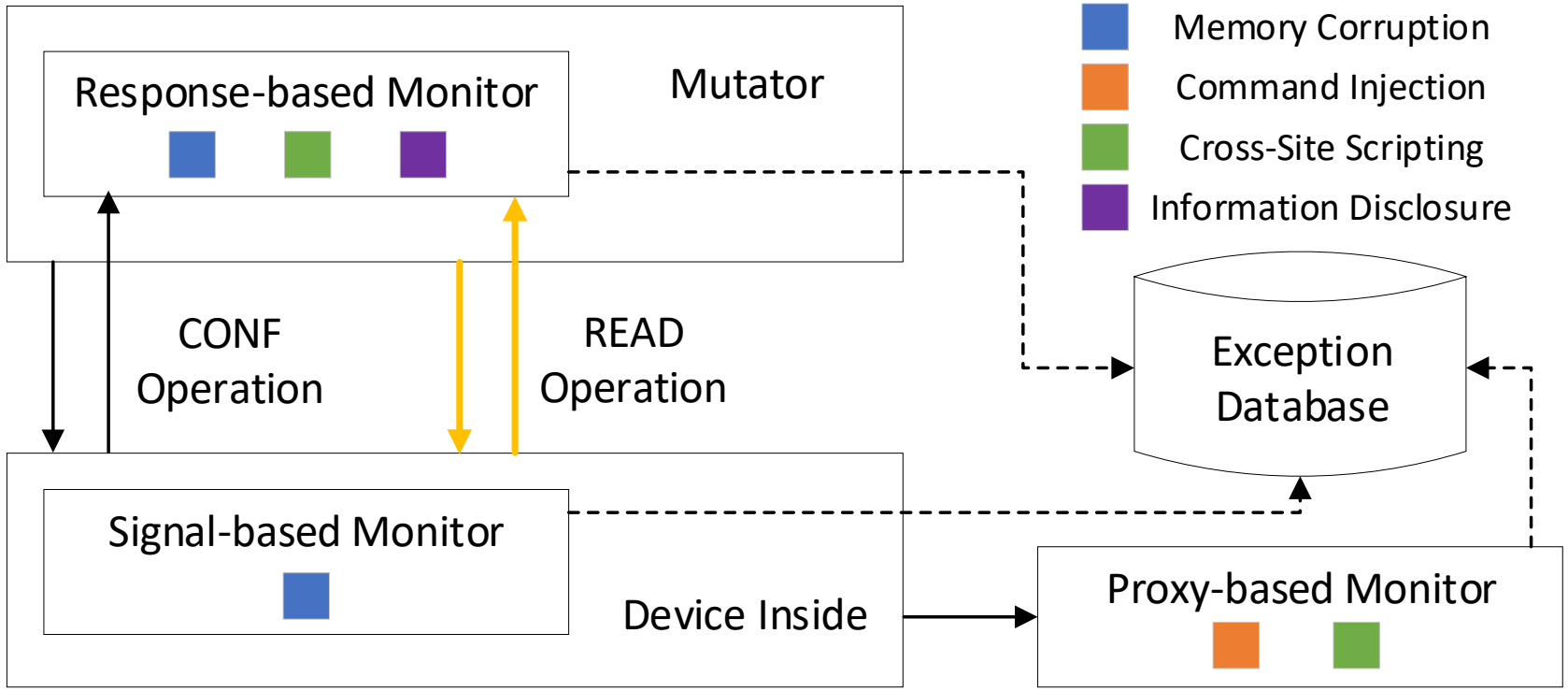
Seed Mutation

Key	Original Value	Attribute	Mutation Rule	Example of Mutated Value
ntpserver1	time.test1.com	variable string	Overflow	time.test1.comtime.test1.com... (repeat 20 times)
			NULL-Pointer dereference	(empty value)
			Command Injection	time.test1.com";wget http://192.168.1.2/ntpserver1;
			Stored XSS	time.test1.com";<script> alert('xss_ntpserver1')</script>
submit_flag	ntp_debug	fixed string	fixed string	ntp_debug
		variable string	...	…
hidden_dstflag	0	number	number	-1
		variable string	…	…

Exceptional Behavior Triggering and Monitoring

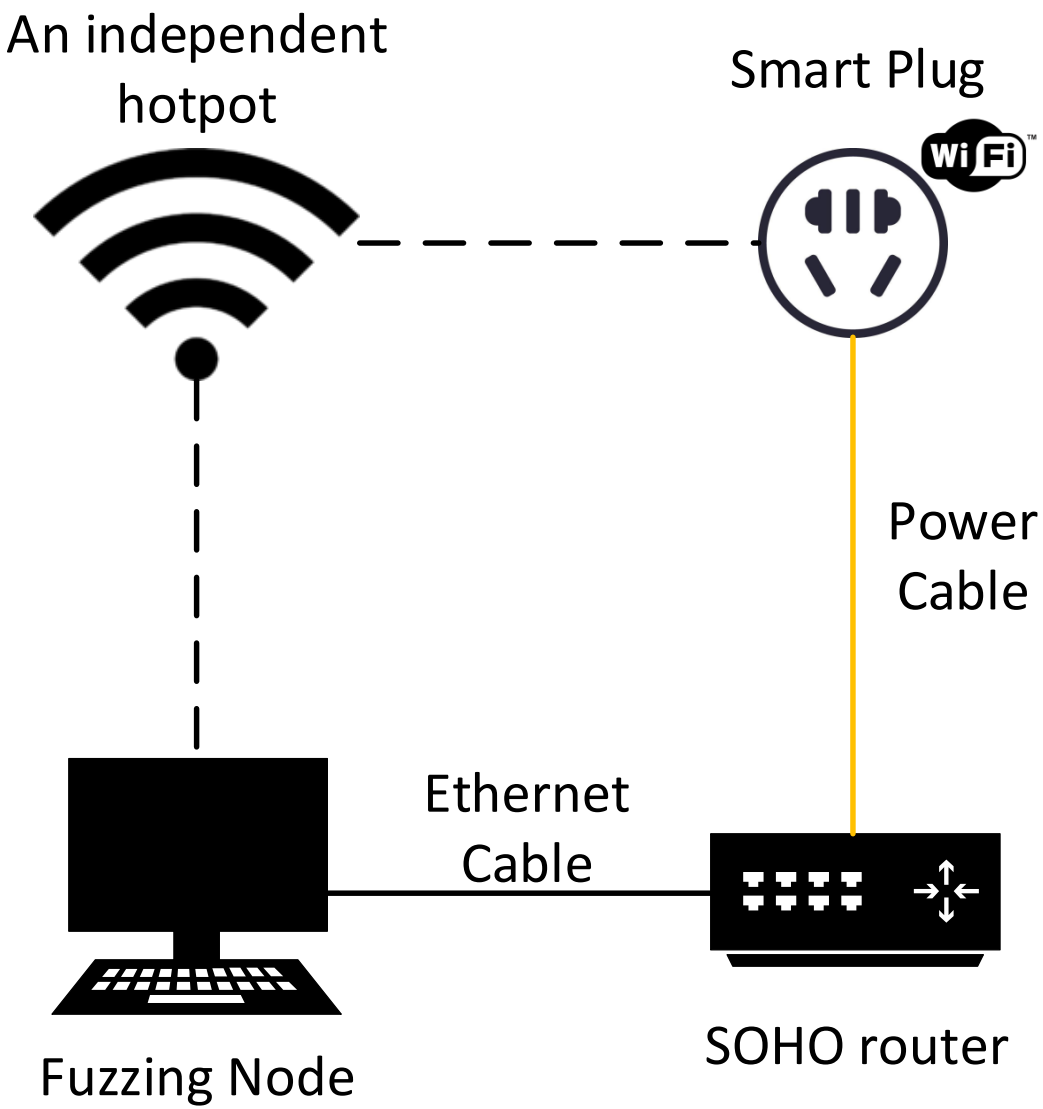
- A CONF operation for the first step
- A READ operation after a CONF operation
- Three typical monitoring mechanisms
  - Response-based monitor
  - Proxy-based monitor
  - Signal-based monitor

这里应该说明如何判定一种漏洞是否触发



Power Control

- Use an extra hotpot to connect the Smart Plug and Fuzzing Node
- Use Mi Smart Plug and python-miio package in practice



Experiment Overview

- We selected 10 devices from 5 different popular vendors
- We obtained 101 unique issues, 97 of which were assigned vulnerability IDs
- We manually crafted the PoCs for all unique issues

ID	Vendor	Product	Firmware Version	Architecture	Signal-based Monitor
1	NETGEAR	Orbi	V15.03.05.19 (6318)_CN	ARM32 (LE)	Device Feature, Serial Port
2	NETGEAR	Insight Managed Smart Cloud Wireless Access Point	WAC505-510_firmware_V5.0.5.4	ARM32 (LE)	Not Support
3	NETGEAR	WNDR-4500v3	WNDR4500v3-V1.0.0.50	MIPS32 (BE)	Device Feature, Serial Port
4	NETGEAR	R8500	R8500-v1.0.2.100, R8500-V1.0.2.116	ARM32 (LE)	Device Feature, Serial Port
5	NETGEAR	R7800	R7800-V1.0.2.44, R7800-V1.0.2.46	ARM32 (LE)	Device Feature, Serial Port
6	TP-Link	TL-WVR900G	V3.0_170306	MIPS32 (BE)	Not Support
7	Mercury	Mer450	MER1200GV1.0	MIPS32 (BE)	Not Support
8	Tenda	G3	V15.11.0.5	ARM32 (LE)	Existed Vulnerability
9	Tenda	AC9	V15.03.05.19	ARM32 (LE)	Existed Vulnerability
10	Asus	RT-AC1200	RT-AC1200-3.0.0.4.380.9880	MIPS32 (LE)	Device Feature

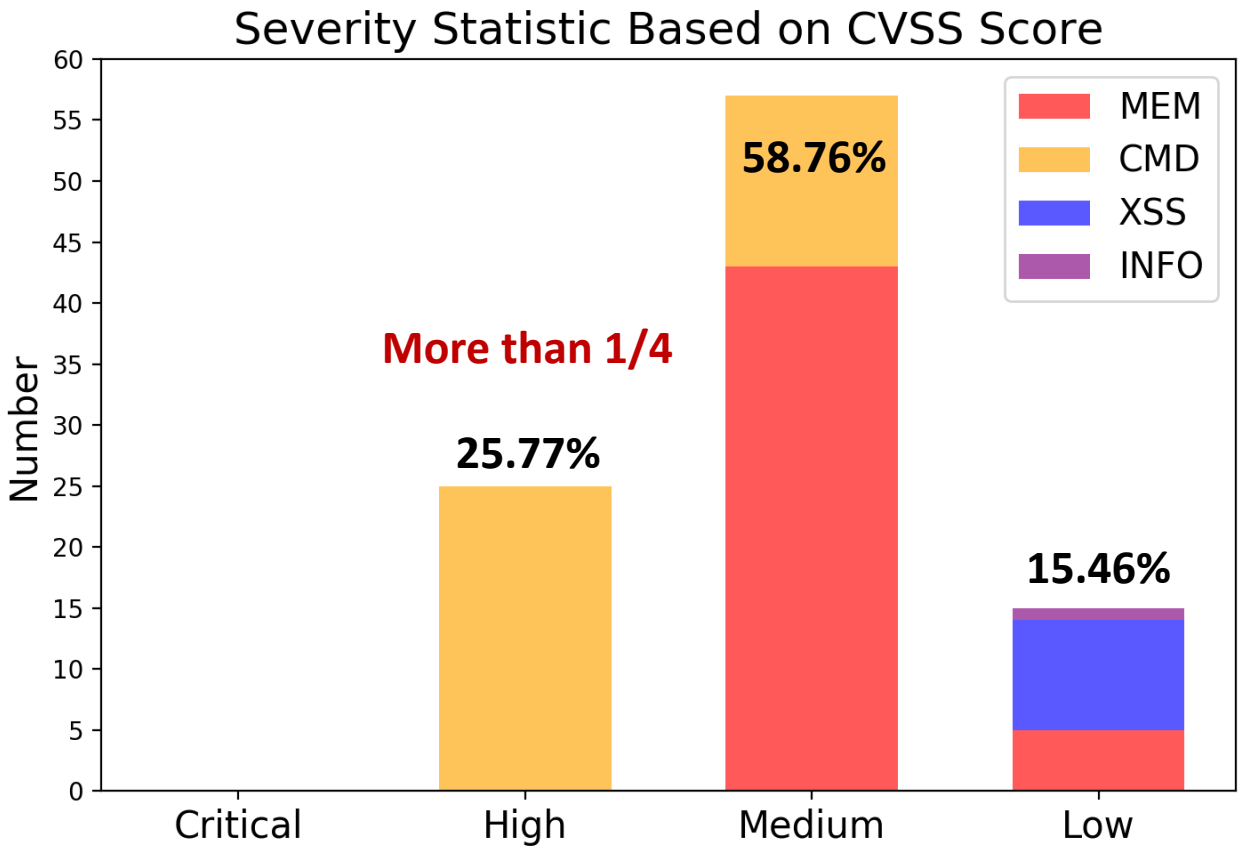
Analysis of Issues

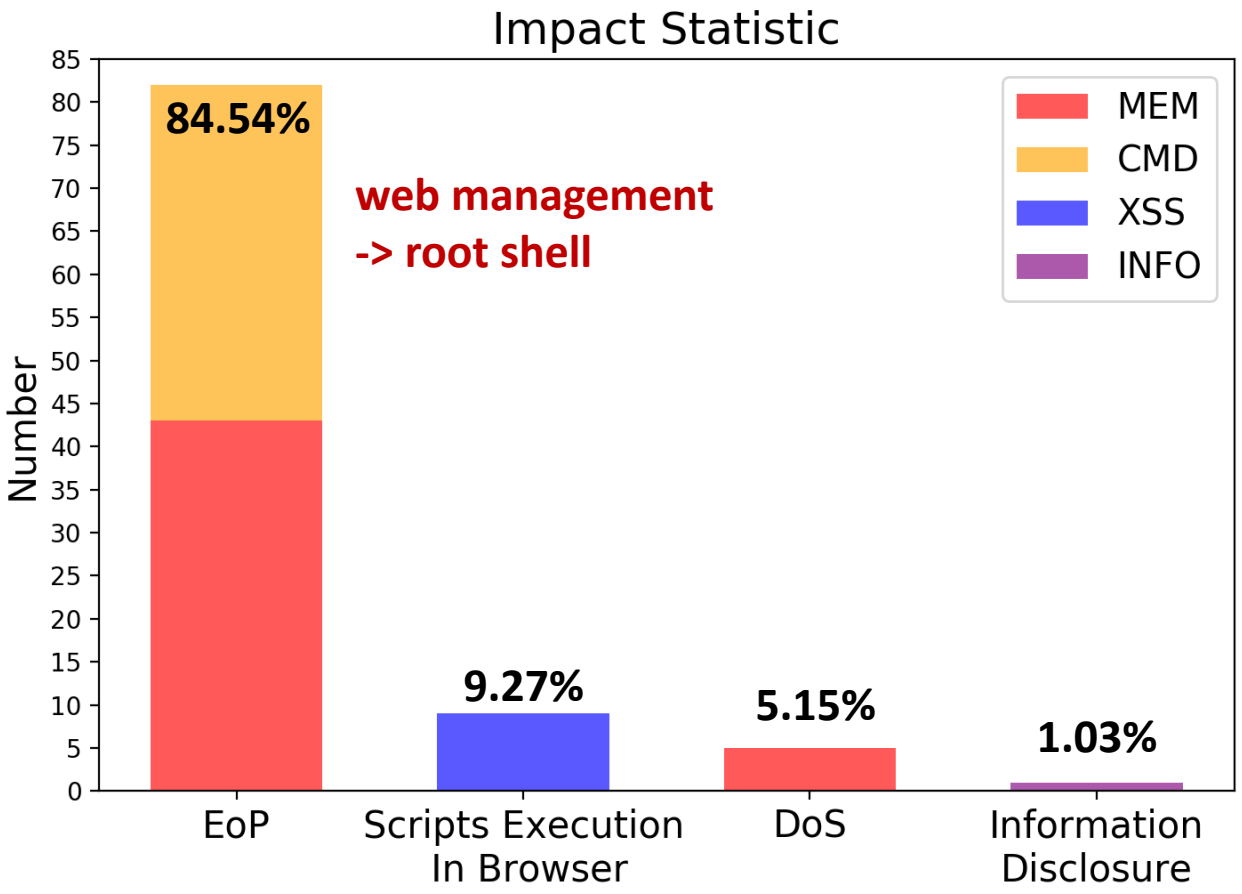
- 101 confirmed issues
  - 48 memory corruption
  - 39 command injection
  - 9 stored XSS
  - 5 info disclosure
- 67.33% are triggered in CONF
- 32.67% are triggered in READ
- Device specificity
  - TP-Link TL- WVR900G

PRODUCT	CONF		READ				SUM
	MEM	CMD	MEM	CMD	XSS	INFO	
Orbi	0	0	0	0	1	1*	2
Insight	0	1	0	0	0	0	1
WNDR- 4500v3	6	2	7	0	0	1*	16
R8500	9	0	0	0	3	1*	13
R7800	0	8	10	0	5	1*	24
TL- WVR900G	0	24	0	1	0	0	25
Mer450	0	2	0	0	0	0	2
G3	5	0	0	0	0	0	5
AC9	11	0	0	1	0	0	12
RT-AC1200	0	0	0	0	0	1	1
SUM	**31**	**37**	**17**	**2**	**9**	**5**	**101**

Analysis of Issues

- 97 assigned IDs (43 CVEs + 52 PSVs + 2 CNVDs)

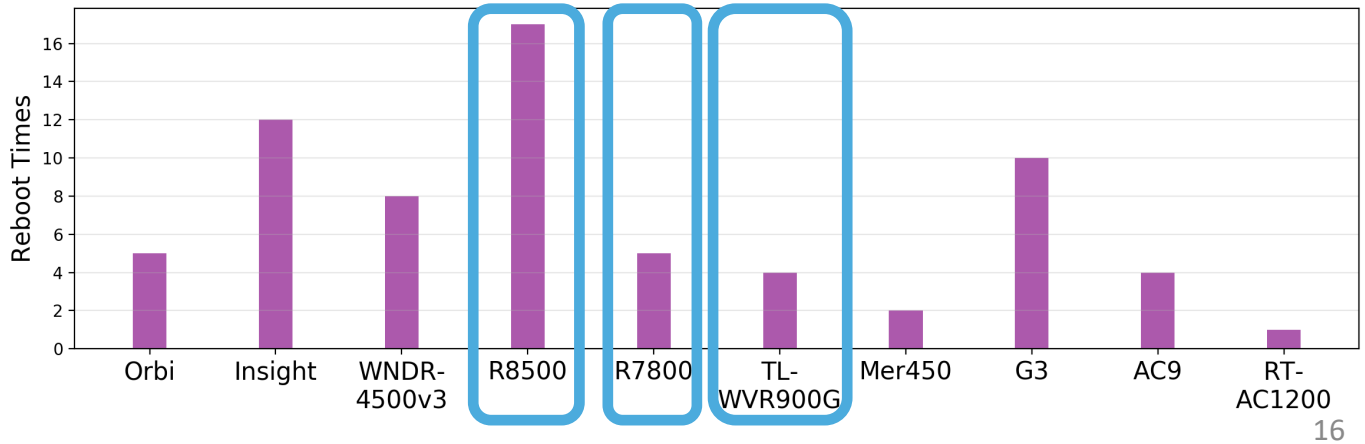




Performance of Monitors

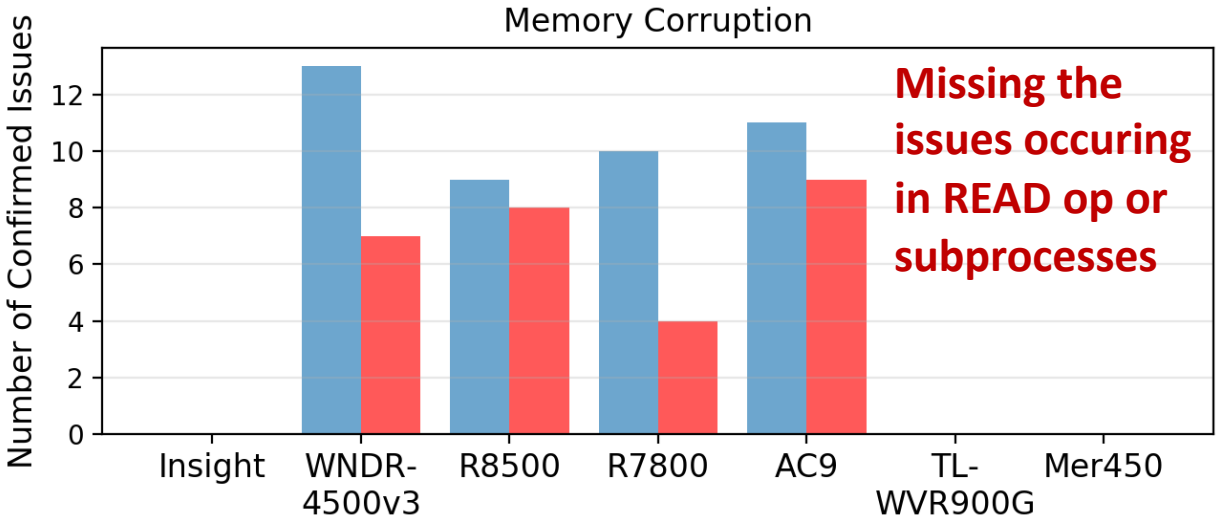
- 77.23% confirmed issues are caught by the general mechanism (response-based and proxy-based monitor).
- Signal-based monitor can catch the silent memory corruption
- Device rebooted 6.8 times on average
  - Handle requests in one process
  - Handle requests in subprocesses
  - Backend is developed on top of OpenWRT

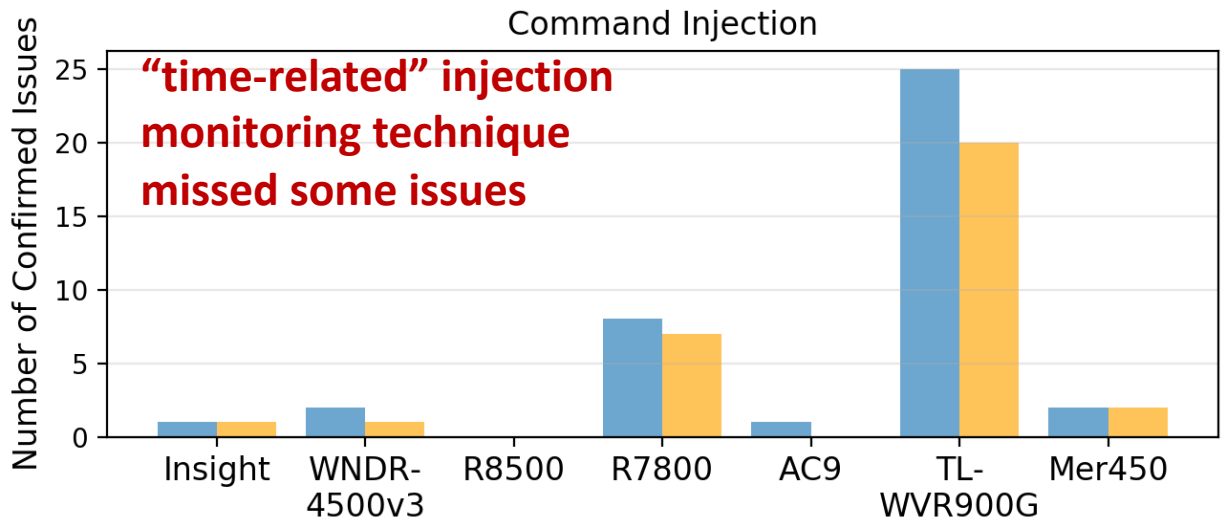
PRODUCT	MEM		CMD	XSS		INFO
	R	S	P	P	R	R
Orbi	0	0	0	1	0	1
Insight	0	N/A	1	0	0	0
WNDR-4500v3	3	10	2	0	0	1
R8500	7	2	0	1	2	1
R7800	2	8	8	2	3	1
TL-WVR900G	0	N/A	25	0	0	0
Mer450	0	N/A	2	0	0	0
G3	4	1	0	0	0	0
AC9	9	2	1	0	0	0
RT-AC1200	0	0	0	0	0	1
SUM	25	23	39	4	5	5

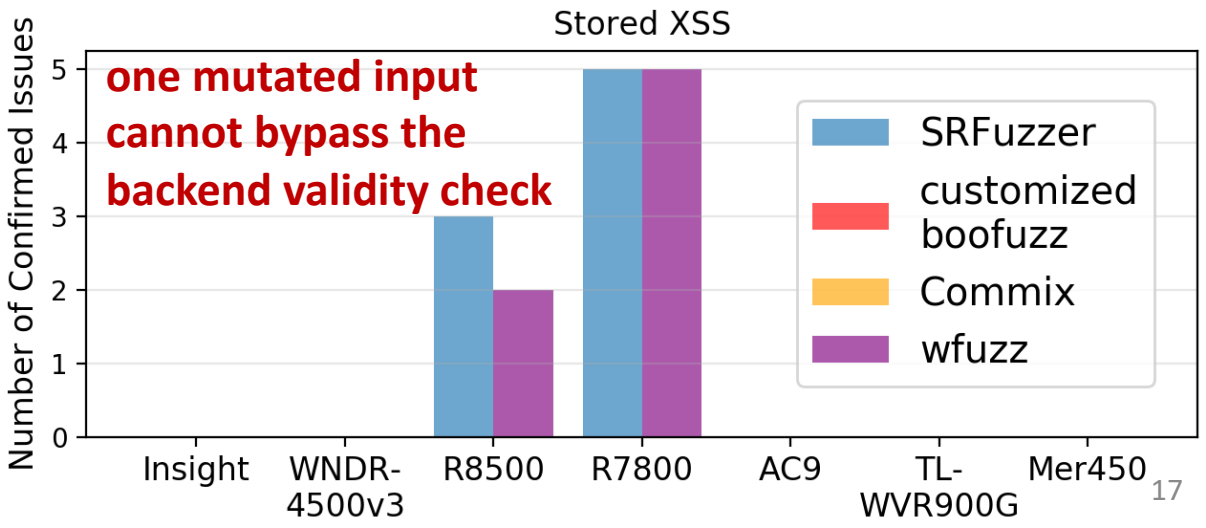


Comparison with Popular Fuzzers

- Selected 7 devices randomly
- More memory corruption issues than customized boofuzz by 53.57%
- More command injection issues than Commix by 25.81%
- Similar performance with wfuzz on XSS detection







Discussion

- Limitation of the scope
  - More types of device
  - More types of protocol
- Vulnerability of severity
  - More critical issues
- Research on data inconsistency
  - To find more issues and help vendors to harden their products.
- Monitoring
  - Make the efficient monitoring mechanism more general

Summary

- We present SRFuzzer for physical SOHO routers to automatically discover multi-type vulnerabilities
- We reveal the root cause of the different types of vulnerability as data inconsistency
- We obtain 97 assigned vulnerability IDs by fuzzing 10 popular real-world devices

REFERENCE

[1] 2018. New VPNFilter malware targets at least 500K networking devices worldwide.

https://blog.talosintelligence.com/2018/05/VPNFilter.html.

THANKS

Q&A