

HVCK

onetwozerotwofive

a celebration of digital counter culture

OFFENSIVE X

redefining what a hacker
conference is all about

hacking the gibson

emulating modern
mainframes

fissure

opensource RF and
reverse engineering
framework

the dons of distortion

ringmasters of the
cyber circus

DUB



68EF5DD4EA03B70F10E1F83651017DF13265EB3



SEC

3C856E567C93EF2848AF2FE704A1211674C3A



I have created so many puzzles and
forget where they all are. I almost
been solved until the Rain came
anew. There are at least 25 cryptic
easter eggs in the previous issues.
unclaimed prizes either way, I am
ever solves the most by Friday 9

and challenges for HVCK that even I
st quit doing it, very few had ever
e. So I throw out my challenge
ic challenges hidden ciphers and
s of HVCK. Each probably still has
n offering a cash prize for whom
th May at 10pm.

\$250
lotsa luv
d8rh8r





See your business
through the eyes of an
adversary

Smart Cyber Solutions: the devil you know

Physical and digital security consulting

Privacy & anonymity solutions

Threat & digital foot print assessments

Pro bono services available for registered charities*



Tox: 49B4FBFE9CE99512564C0046AE55799E5ACD90C50EE2C233030AC4A295505307AFFED7D94571



OFFEN

Athens
Conservatoire

An illustration of three men wearing large headphones, looking intently at something off-camera. The man on the left has dark hair and glasses, the middle man has short spiky hair and glasses, and the man on the right has long hair and glasses. They are in a dark, blue-toned environment with glowing lines and a digital aesthetic. The text 'SIVE X' is overlaid in a stylized, glitchy font.

SIVE X

June¹⁸⁻¹⁹
2025

A girl's best friend.

d8rh8r - Hans Groover (Nakatomi Mix)



D8RH8R

HANSCHLOVER

Leading Value: 15.034 pt

Margins

Top: 75.169 pt Inside: 53.15 pt

Bottom: 60.135 pt Outside: 53.15 pt

HERO SMITH

Columns → Number: 8 | Gutter: 7.5 pt

Row → Number: 12 | Gutter: 5.304 pt

Hero Smith for Waterfront Productions Limited and Contemporary Communications Corp.

Secondary → Number: 2 | Gutter: 7.5 pt



Letters from the editor

Written by **D8RH8R**

Hello, you beautiful chaos agents of the cyber realm—it's Call for Papers season, that glorious stretch where the infosec community sharpens its pencils, cracks its knuckles, and floods the conference circuit with ideas ranging from genius to “did you write this at 3 a.m.?”

Welcome to our latest HVCK Magazine, where we're wading into the madness with a twist. This year, I'm ditching the heavy-handed location themed doom & gloom sermonising for something a little more... cinematic. I've tossed my hat in the CFP ring with a pair of talks

imagining how Hans Gruber—that suave, steely-eyed bastard from Die Hard—would pull off his iconic heist if he was flat broke.

Picture it: no fat stacks of cash to fund his Nakatomi Plaza takeover, just a guy with a velvet voice, a thrift-store suit, and a budget tighter than airport

security. How does Hans storm the tower with nothing but pocket lint and spite? This issue also sees our first takeover. OffensiveX is an security conference in Athens that we believe are doing all the right things as far as conferences go. Now if could only get to Greece for June.

**A short
note from
d8rh8r:**

**Here at HVCK we have
always preached that we
back the people, companies,
and events we believe in.**

**OffensiveX is all three.
Here's how they won us over. . .**

**Speakers, not sponsors are
the focus of their event and
are treated as such..**

**CFP was based on the merit
of the paper. Nothing else
entered the discussion.**

**A complimentary OffensiveX
badge waits at the entrance
for all women in cyber**

Legends.



**PLACE
INVADER**
CYBER AND
PHYSICAL
PENTESTING



100

com



An
interview
with

**Kev
Muir**

Prologue:

The Beast in the Basement

The air hung thick and cold, smelling faintly of ozone and dust. Not the sterile, antiseptic smell of a modern data centre, but something older, more... primal. This wasn't a server farm; this was a mainframe lair. Rows of black monoliths hummed with a low, guttural throb, a pulse that seemed to vibrate in my bones. I was here, 6 years old, deep in the belly of the beast. Wargames had been the catalyst for this rare visit to my old man's work. I wouldn't encounter another mainframe until 7 years later. Funnily enough, my next encounter began at the same place my first had: the video store, and its name would be Gibson.

"They're not much to look at," agreed Kev over the LinkedIn chat to organise this very article. "Trust me, these things are incredible. They're responsible for processing the world's transactions at lightning speeds. Be it votes for the government, tracking stock levels for global retailers, booking flights for a weekend away or ensuring your pay goes in on time, mainframes make all this possible and keep the modern world moving."

I opened my notebook. "Okay, Kev, let's start there. 'Mainframe' – for most people, it conjures up dusty images, maybe punch cards, NASA control rooms. What are we actually talking about these days, in 2025?"

"Forget the movies. Forget the outdated stereotypes. A mainframe today is the ultimate specialised server. It's not your web-browsing, game-playing machine. It's a goddamn workhorse. We're talking massive throughput, rock-solid reliability, and security that's tighter than Fort Knox."

"Right, so not something you pick up down at JB Hi-Fi with the flat-screens?" I asked.

ell no. These things are built to run 24/7/365, for years without a hiccup. Redundancy is the name of the game. Multiple everything. If one part fails, another kicks in before you can blink. Think about it: if a mainframe goes down, the world feels it. Banks freeze, planes get grounded, chaos ensues.”

“Okay, powerful, reliable... So what’s under the hood? What OS drives this beast? Not Windows, I’m guessing.”

Chapter 2: z/OS: Welcome to the Matrix

“z/OS,” Kev said. “IBM’s mainframe operating system. It’s a whole different universe from anything you’ve seen. Windows, Linux... those are toys compared to this. z/OS is built from the ground up for one thing: moving mountains of data, securely and efficiently.”

“So, we’re talking no graphical interface? No familiar point-and-click comfort zone here?”

“Forget it. This is the command-line kingdom. We’re talking text-based interfaces, cryptic commands, and a learning curve that’s steeper than Everest. A GUI? That’s just wasted resources. This is a server OS, and like most server-based *nix systems, a GUI is bloatware.”

An image popped up in our chat window. “This is your portal: TSO/ISPF.”

“TSO/ISPF,” I repeated, the acronyms sounding pretty arcane. “Can you break that down a bit?”

“Time Sharing Option/Interactive System Productivity Facility. TSO is the engine, the command interpreter. ISPF is a layer on top, a menu-driven system that makes it slightly less terrifying. Slightly. Think of TSO as bash or PowerShell, then ISPF is a series of text menus. It’s all keyboard, all the time. No mouse jockeys here.”

Chapter 3: Datasets: Not Your Grandma’s File System

“So how does this beast actually handle data?” I asked. “Is it just normal files and folders like we’re used to, or something different?”

“Nothing about a mainframe is ‘normal.’ We’re talking ‘datasets.’ These aren’t just files; they’re structured entities. You define the record length, the format, the block size... everything. It’s all about squeezing every last drop of performance and ensuring data integrity.”

He rattled off a string of acronyms: “PS, PDS, VSAM... Physical Sequential, Partitioned Data Set, Virtual Storage Access Method. Each one has its purpose. PS is your basic text file. PDS is like a directory, holding multiple ‘members’ – individual files. VSAM is your database-style workhorse, with indexing and keyed access.”

“Okay, so it’s a highly controlled, almost regimented way of organising data? Precision engineering.”

“Exactly. This isn’t your desktop clutter. This is a finely tuned orchestra, where every instrument has its place and plays

its part perfectly,” Kev crowed, clearly passionate about the structure.

Chapter 4: Security: Lockdown Level – Maximum

“Alright, let’s talk security. Given what these machines are doing, you’d assume it’s Fort Knox level, yeah?” I said, leaning in slightly.

“Fort Knox is an understatement,” Kev said, a glint in his eye. “Mainframe security is built on layers upon layers of access control. It starts with the basics, user IDs, passwords, but it goes way, way beyond that.”

“So how does that work on the user side? How are individuals managed in this high-security environment?”

“Every user gets a unique ID, handed out by a system admin. These IDs are linked to profiles, which are basically blueprints of what that user can do. Can they read this dataset? Can they execute that command? Can they access this subsystem? It’s all defined in the profile. And the whole thing is typically managed by one of three behemoths: RACF, ACF2, or Top Secret.”

“They’re the gatekeepers, the enforcers,” Kev confirmed.

“They handle authentication,

authorisation, and auditing. They verify who you are, decide what you can access, and track every damn thing you do.”

“So these profiles... they function like permissions?”

“More like security clearances. They’re granular. You can control access down to the individual dataset, the individual command, the individual transaction. And you can group them into roles – developers, operators, auditors – each with their own set of permissions. It’s all about least privilege: give people only what they need, and nothing more.”

Chapter 5: Hunting for Weaknesses: A Pentester’s Playground

“But even Fort Knox has weak points, right? Like in Hackers with Gibson. Nothing’s impenetrable. Where do things tend to go wrong? Where do the mistakes creep in?”

Kev’s smile turned predatory. “The same place they always do: human error. Overly broad permissions are the classic blunder. Giving a developer full access to production data when they only need read access for testing. Weak passwords are still a problem, even here. Misconfigured access controls, where

Data Set

```
-----  
SYS1.SBLSTB  
SYS1.SBPXTE  
SYS1.SCBDTE  
GIM.SGIMTEN  
ISF.SISFTLI  
SYS1.SERBT  
SYS1.SERBTE  
SYS1.SMP.OT  
JSZ.SPFLOG1  
ISF.SISFPLI  
SYS1.SMP.OT  
P.DYS300.QA  
P.DYS300.MA  
P.DYS300.RE  
TSC.PROJ.AC  
ISP.SISPEXE  
SYS1.SBPXEX  
SYS2.SYB.R2  
SYS1.HELP  
ISP.SISPHEL  
NULLFILE  
SYS1.BRODCA  
NULLFILE  
JSZ.TEST.CL  
SYS2.USER.C  
SYS2.CLIST  
SYS2.TONE.C  
SYS2.PDS84.  
SCLM.EXEC  
ADCD.DUMMY  
ISP.SISPCLI  
SYS1.DGTCLI
```

L0
NU
NU
U
B

NU
ABLES
.LIST
B
ABLES
.LOAD
INT.LOAD
LEASE.LOAD
TIVE.CUST
C
EC
10.EXEC

P

ST

IST
LIST

LIST
ASM

B
B

permissions are just plain wrong. Unencrypted protocols, like using Telnet instead of SSH... it's like leaving the front door unlocked."

"And what about legacy code? That's often a minefield in other areas, isn't it?" I asked.

"Bingo. Old code, written before security was a top priority, can be riddled with holes. And don't forget about insecurely stored encryption keys. If those get compromised, all your fancy encryption is worthless."

"Okay, hypothetically... If someone was trying to find a way in, like those kids heading for Grand Central... where would the likely starting points be?"

"TSO/ISPF itself can be a playground," Kev said. "If it's not configured properly, you might find ways to escalate privileges, execute hidden commands... And then there's z/OS UNIX System Services – USS. That's a whole Unix-like environment running inside the mainframe. And guess what? Hackers know Unix."

Chapter 6: The Power Tools: APF, JCL, and the Scripting Underworld

I scanned my notes. "Right, APF-authorized programs. That

sounds... significant. What's the story there?"

"APF – Authorised Program Facility – is where things get really serious," Kev explained. "APF-authorized programs run with god-like privileges. They bypass normal security checks. They have access to resources that are usually off-limits. These are the core system programs, the ones that need to operate outside the usual rules."

"So, if you manage to get malicious code into an APF library... that's pretty much game over?"

"You own the box," Kev finished, his voice low. "That's why those libraries are guarded like the crown jewels. They're the VIP lounge of the mainframe."

"JCL... Job Control Language. Sounds fairly fundamental to making things happen."

"It is. JCL is how you tell the mainframe what to do. It's the scripting language for batch jobs. You write a JCL script to tell it which program to run, which datasets to use, how much memory to allocate... everything."



"And controlling the JCL means... controlling the workflow, the core processes?"

"You can control the mainframe. You can run unauthorised programs, access sensitive data, delete anything you want. It's like having root access and being able to schedule anything you want with cron."

"And REXX and CLIST? Are they similar tools for automation, just different flavours?"

"Yep. REXX is the modern, powerful one, like Python. CLIST is older, simpler, like a DOS batch file. They're used to automate tasks within TSO/ISPF. And, just like JCL, if they're not secured properly, they can be used to wreak havoc."

Chapter 7: Subsystems and Networks: The Wider Attack Surface

Looking at my notes again. "Okay, these subsystems – CICS, IMS, DB2 – you hear those names a lot in relation to mainframes. What roles do they play?"

"Those are the heavy hitters," Kev said. "CICS – Customer Information Control System – is your transaction processing engine. Think banking, airline reservations, anything that needs to handle a constant stream of requests. IMS – Information Management System – is a hierarchical database, used for massive, complex datasets. And DB2 is a relational database, like Oracle or SQL Server."

"And presumably, each one presents its own attack surface, its own set of potential vulnerabilities?"

"Absolutely. CICS applications can be vulnerable to SQL injection, cross-site scripting, the usual web app stuff, especially now that everything's hooked up to CICS using APIs. IMS needs to be locked down to protect its hierarchical data. And DB2, like any database, is vulnerable to SQL injection and other database-specific attacks."

"How do these systems typically talk to the outside world these days? What does the networking look like?"

"Traditionally, mainframes used their own proprietary protocols, like SNA," Kev explained. "But these days, it's mostly TCP/IP, just like everything else. The networking is robust, but, again, old protocols, open ports... those are your weak points. And remember, mainframes often use EBCDIC for character encoding, not ASCII. So, if you're sniffing network traffic, it'll look like garbage unless you have a tool to decode it."

Chapter 8: z/OS UNIX: The Trojan Horse?

"Interesting. z/OS UNIX System Services... USS. That sounds like a bridge to more familiar territory, perhaps?"

"It is, in a way," Kev agreed. "USS provides a Unix-like environment within z/OS. It lets you run Unix applications and use familiar Unix tools. It's a bridge between the old world and the new."

"Which, naturally, makes it another potential vector for trouble if not managed carefully?"

```

kev@DESKTOP-Q69AF9M: ~
File      Options  Keypad

-----
THIS SYSTEM CONTAINS RESTRICTED INFORMATION FOR AUTHORISED USERS ONLY
THE SYSTEM MAY BE MONITORED, RECORDED, AND IS SUBJECT TO AUDIT
UNAUTHORIZED USE OF THE SYSTEM IS PROHIBITED UNDER SCOTTISH LAW.
USE OF THE SYSTEM INDICATES CONSENT TO MONITORING AND RECORDING.
-----

          SSSSSS      BBBBBS
        SS          BB  BB
        SS          BB  BB
        SSSSSSS     BBBBBS
          SS        BB  BB
          SS        BB  BB
        SSSSSSS  0  BBBBBS

      LU NAME:  LU296                      IP ADDRESS: 127.0.0.1

      SENSE CODE: AC/00                     PORT:41194

-----
      eZOSv2.2                      eZICS v1.2                      eZDB v1.2

LOGON using L TSO, L CICS or L DB2.
Logon Type:  █
█A█
024/013

```

```

kev@DESKTOP-Q69AF9M: ~/mfsim/editor
eZedit - 1.00  LONGER.TEXT                      COLUMNS 7 3
Command ==>                                     Scroll ==> CSR
***** ZAP*****AUTOSAVE***** TOP OF DATA *****
''''''#!/usr/bin/env python3
''''''import socket
''''''import threading
j2█ import time
''''''import os
''''''import argparse
''''''import logging
''''''from logging.handlers import RotatingFileHandler
''''''from datetime import datetime
''''''from passlib.hash import md5_crypt
''''''import subprocess
''''''import select
''''''import pty
''''''import re
''''''
''''''# Global dictionary to track active sessions by username.
''''''active_sessions = {}
''''''
''''''# =====
''''''# MD5CRYPT FUNCTIONS (weaker hashing for lab use)
***** ZAP*****AUTOSAVE***** BOTTOM OF DATA ***** 6928K FREE

```

“Exactly. It’s another attack surface. While USS is integrated with z/OS security, it also brings in all the usual Unix vulnerabilities. It’s something that needs to be carefully managed and secured.”

Chapter 9: Dynamic LPA and the Assembler Enigma

“What else?” Kev pondered. “Oh, you can update system code on the fly with Dynamic LPA, without having to restart the system. Great for patching, but also a huge security risk. If an attacker can inject code into Dynamic LPA, they can potentially take control.”

“I keep hearing about Assembler. Is knowing that low-level language still considered the key to really understanding the mainframe at its deepest level, or is that changing?”

“Assembler is the native language of the mainframe,” Kev said. “It’s low-level, close to the metal. Understanding assembler lets you dissect system code, find hidden vulnerabilities, and even write your own exploits. It’s like having X-ray vision into the machine’s soul. But, with today’s AI, it’s not essential for effective penetration testing, which is good, as there are fewer and fewer of us assembler geeks around.”

Chapter 10: eZmainframe: The Genesis of a Renegade Project

“Okay, Kev, this brings us to eZmainframe. This is your baby, right? Tell me about it – what is it, and what drove you to create it?”

Kev’s eyes lit up, the weariness from the technical deep-dive replaced by a spark of passion. “eZmainframe is my attempt to democratise mainframe knowledge. It’s a free, open-source mainframe simulator. It’s not a perfect emulation of a full z/OS environment – that would be impossible. But it’s designed to be a learning tool, a playground, a honeypot... for anyone who wants to get their hands dirty with mainframe technology without needing a multi-million dollar budget.”

“So, was it born partly out of frustration with the lack of access to these systems?”

“Damn right. Try getting your hands on a real mainframe. It’s like trying to get into Area 51. They’re expensive, they’re locked down, and you can’t just experiment on a production system that’s processing billions of dollars in transactions. Even the development instances are pricey. And the existing open-source emulator, Hercules, doesn’t fully simulate the security aspects, like RACF or the Unix subsystem.”

“So, you essentially decided if you couldn’t get easy access to the real thing, you’d build your own accessible playground?”

“I had to. I needed a platform for teaching, for research, for my own sanity. Something I could break, exploit, and then share with the world. Something that responded to security scans like a real mainframe, that let you play with TSO/ISPF, that gave you a taste of the real thing, without the risk.”

Chapter 11: Python, Design, and the Future: Building a Community

“Interesting choice – Python. Not exactly the first language people associate with mainframes. Why Python for this project?”

Kev grinned. “I’m not a coding god. I’m an infrastructure guy. Python is relatively easy to learn, it’s got great libraries, and, frankly, AI can help me fix my screw-ups. It might get rewritten in something else later, but for now, Python is perfect for rapid prototyping and getting things done.”

“How did you even approach simulating something as complex as z/OS? Where do you start making those design choices?”

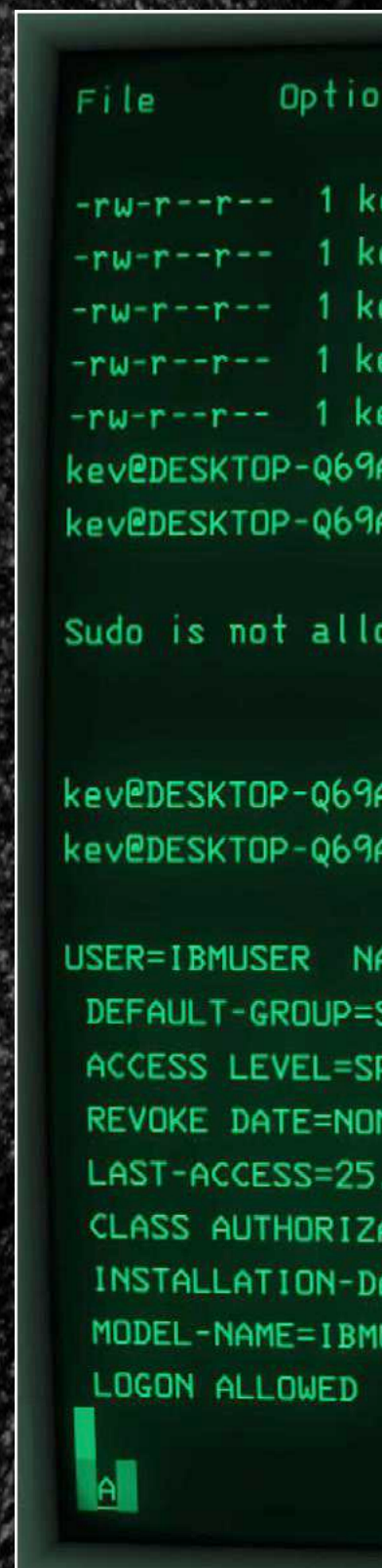
“Carefully,” Kev said, with a laugh. “I focused on the essentials: the core TSO commands, the menu structure, the feel of the interface. I wanted it to respond like a mainframe, even if the underlying mechanics were different. I used Linux to mimic the Unix subsystem and some of the RACF commands. It’s all smoke and mirrors, really, but it’s convincing smoke and mirrors.”

“And it’s completely open source? What’s the vision for the community around it?”

“Completely. I want people to use it, to contribute to it, to build a community around it. It’s a work in progress, always evolving. I’m using it myself to build a lab for my ‘Hacking Mainframes’ book and for training courses. It’s a tool for learning, for breaking things, and, hopefully, for making the mainframe world a little less intimidating.”

Epilogue: Beyond the Black Boxes

I closed the chat. The office felt quiet after the deep dive. Kev had shone a light into the intimidating world of the black box. With eZmainframe, maybe that light could spread, make it less of a fortress on the hill. The walls still stood, tall and imposing, but perhaps the adventure of understanding them was just beginning for a wider audience.



ns Keypad

```
ev kev            62 Feb 12 01:39 MYJOB.JCL
ev kev 14008320 Feb 13 23:46 RACF
ev kev 14008320 Feb 11 00:04 SYS1.RACFDS
ev kev            7 Feb 12 01:41 TESTING
ev kev            537 Feb 11 00:04 time.rexx
```

AF9M:~/mfsim/f/IBMUSER\$

AF9M:~/mfsim/f/IBMUSER\$ sudo

owed in OMVS shell.

AF9M:~/mfsim/f/IBMUSER\$

AF9M:~/mfsim/f/IBMUSER\$ TSOCMD LISTUSER

```
NAME=IBMUSER                    OWNER=#SYSPROG    CREATED=24.271
SYS1            PASSDATE=25.020 PASS-INTERVAL= 30 PHRASEDATE=N/A
SPECIAL
NE    RESUME DATE=NONE
.041/00:09:58
ATIONS=NONE
ATA=DEADLY DIGITS OF DEATH
USER
```

(DAYS)

(TIME)

024/001



asymmetric
apparel • artist • audio





asymmetric-1337.teemill.com
asymmetric-1337.teemill.com

Lifting the bar on what a offensive security conference should be.

HVCK has a history of getting behind companies, events and people we truly believe in. Another has entered the fold. OffensiveX are throwing conferences the way they should be, with the prestige being returned to the speakers rather than the vendors.

Providing a true experience for those dedicated enough to submit a paper that gets accepted, speakers lucky enough to grace the stage this year will not only be catching 40 under 5 stars but can rest easy in the fact that their personal guide and driver will make sure they're on time to cast off on the speakers decompression day, yachting on that beautiful azure water..

Its the kind of event HVCK would be proud to throw and as a hacker would love to go. I have a sneaking suspicion this wont be the last time you hear about OffensiveX in HVCK Magazine.

OFFEN



SIVEX



18-19
June
2025

The Athens Conservatoire

In the marble-walled halls of Athens' historic Conservatoire, a different kind of symphony will play out this June — one composed of shellcode, payloads, and packet captures. Offensive X is back. And it's more loaded than ever.

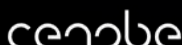
Forget stuffy vendor expos and rinse-and-repeat talks. Offensive X is a full-throttle hacker conference built for practitioners, red teamers, reverse engineers, and exploit developers. Two days of brutal technical content, underground village demos, and a no-BS approach to cybersecurity education.

The Agenda: Break First, Ask Later

Offensive X is built for people who do the work — red teamers, malware analysts, exploit devs, and adversarial thinkers. The 2025 edition kicks it up a notch, diving headfirst into:

- Advanced Red Team TTPs
- Active Directory and Azure AD Attacks
- AI and LLM Exploitation
- Malware Reverse Engineering
- IoT/Automotive/Embedded Offence
- ICS and Industrial Control System Threats
- Windows EDR Evasion & LOLBAS
- Supply Chain and Firmware Intrusions

There's no hand-holding here — just surgical techniques, direct application, and bleeding-edge demos that will mess with your blue team's sleep cycle.





“*OffensiveX is more than just a conference;
it’s a destination.*

*A place for real connections,
deep technical talks, and moments that actually matter.*

*We’re here to push the limits of security while
keeping it real, technical, and fun—the kind of fun
you actually remember :)*

*Our vision is to create a powerful community,
where we learn, grow, and innovate together.*

”

- Athena Ntova

Co-Founder, **OffensiveX**



“For me, OffensiveX is about creating the kind of space I wish existed when I started out—a place where technical depth meets real collaboration, without the corporate polish or shallow hype. We’re doing this to bring together people who genuinely care about offensive security, not just for the headlines, but for the craft. The goal is to build a community that values hands-on research, open sharing, and honest conversations. If someone leaves the conference with one solid connection, one idea they can build on, or one spark that fuels their curiosity—that’s a win.”

- **Kypriano Vasilopoulos.**

Co-Founder and Tech Lead, **Offensive X**

Keynotes



Dr. Yiannis Pavlosoglou

@KibernaSecurity

Dr. Yiannis Pavlosoglou is a cybersecurity executive with a deep-seated expertise in steering organizations towards robust digital security. He is currently Vice Governor of the National Cybersecurity Authority of Greece.

With over 20 years of experience in Information Security, he has a track record history of contributing to and successfully applying industry-standard frameworks, the latest of is the Artificial Intelligence Resilience Maturity Model (AI-RMM), for which he is a co-author. His technical journey is marked by a PhD in secure routing protocols, over five years as a professional penetration tester, and significant contributions to the public domain, including over 10,000 lines of open source code.



Renaud Feil

@Synaktiv

CEO | Co-Founder of Synaktiv

From the very start of his career, Renaud has been driven by a relentless passion for technical challenges. The harder the problem, the more fun it is to break.

Following this mantra, Synaktiv has grown into a team of 170+ security experts, specializing in Red Team operations, reverse engineering, vulnerability research, security tool development, and incident response.

Renaud may have swapped CTF competitions for customer meetings, but his hacker spirit remains intact. He still believes cybersecurity is the most fascinating field mankind has ever created and continues to lead Synaktiv with the same curiosity, mischief, and love for breaking things that got him into hacking in the first place.

Speakers



Will Schroeder

@Harmj0y

Will Schroeder is a member of the R&D team at SpecterOps, where he helps research and develop new offensive techniques and capabilities. He has spoken at a number of industry conferences including Black Hat and DEF CON on topics spanning AV-evasion, Active Directory, post-exploitation, red team tradecraft, BloodHound, malicious access control, malware, and offensive PowerShell. He is also the cofounder of numerous open source projects including Empire, BloodHound, GhostPack, and more. Will Schroeder



Lee Chagolla-Christensen

@tifkin_

Lee Chagolla-Christensen is a member of the R&D team at SpecterOps, where he helps research and develop new offensive techniques and capabilities. He has an extensive background in offensive security, particularly enjoying research of Windows, Active Directory, and the components commonly found inside them. His research has resulted in several CVEs and new offensive tradecraft used throughout the industry. In addition, Lee has contributed to many open-source tools including GhostPack, BloodHound, SpoolSample, UnmanagedPowerShell, and KeeThief.

Speakers



Rio Sherri

@0x09AL

Rio Sherri is an offensive security professional with over 10 years of experience in the field. He is currently a member of the NetSPI Red Team, where he utilizes his expertise in adversary simulation to improve the security of organizations. He has made significant contributions to the community by developing open-source offensive security tools that are used by many organizations to simulate attacks and strengthen their security posture. Another area of interest for Rio is exploitation and reverse engineering, where he has discovered several vulnerabilities in world leading vendors like Google, Microsoft, Spotify, Symantec, Imperva and many more. In addition to his technical pursuits, he actively contributes to the security community by organizing a range of events, including workshops, Capture the Flag (CTF) competitions, and a local BSides conference. Renaud Feil



Charles Hamilton

@MrUn1k0d3r

Charles Hamilton is a Red Teamer with over twelve years of experience delivering offensive testing services for various clients. In recent years, Charles has specialized in covert Red Team operations targeting complex and highly secured environments. These operations have enabled him to refine his skills in stealthily navigating client networks without detection. Since 2014, he has been the founder and operator of the RingZero Team website, a platform dedicated to teaching hacking fundamentals. Charles is also a prolific toolsmith and trainer who has delivered this training more than 20 times, both online and onsite. He is a speaker in the InfoSec industry, known under the handle Mr.Un1k0d3r.

Speakers



Amat Cama

@amatcama

Amat Cama is an independent security researcher specializing in reverse engineering, vulnerability research, and exploit development. With a deep interest in low-level software, he gained recognition through his success in hacking competitions like Pwn2Own. He has worked at security research companies, contributing to various aspects of the vulnerability research process.



Dirk-Jan Mollema

@_dirkjan

Dirk-Jan Mollema is a security researcher focusing on Active Directory and Microsoft Entra (Azure AD) security. In 2022 he started his own company, Outsider Security, where he performs penetration tests and reviews of enterprise networks and cloud environments. He blogs at dirkjanm.io, where he publishes his research, and shares updates on the many open source security tools he has written over the years. He presented previously at TROOPERS, DEF CON, Black Hat and BlueHat, is a current Microsoft MVP and has been awarded as one of Microsoft's Most Valuable Researchers multiple times.

Speakers



Oddvar Moe

@oddvarmoe

Oddvar has over 24 years of experience in the IT industry, specializing in Windows Security. As a seasoned Red Teamer working for TrustedSec, he performs engagements with Fortune 100 companies on a regular basis and through that gaining insights into some of the world's most secure environments. Oddvar is a Microsoft Most Valuable Professional (MVP) in Security for nine consecutive years, recognized for his expertise and contributions to the security community. He has spoken at leading conferences, including DerbyCon, Paranoia, HackCon, and Nordic Infrastructure Conference, and is known for his work on LOLBins/LOLBAS, the Ultimate AppLocker Bypasslist and discovery of Pre2k accounts in Active Directory. Additionally, he has discovered multiple Windows OS vulnerabilities (CVEs) and persistence techniques adopted by APT groups.



Matthijs Gielen

Matthijs Gielen is a cybersecurity professional with over 10 years of experience. He initially worked in cyber forensics before moving on to red teaming. His expertise lies in red team automation. In his personal life, Matthijs enjoys reading science fiction books, (board) gaming and snowboarding.

Speakers



Siddhant Prakash

@siddhubadhe

Siddhant Prakash is a Security Researcher with core interest in Vulnerability Research, Penetration Testing, and Malware Analysis - 9 years total experience



Leonidas Tsaousis

@laripping

Leo is a Senior Security Consultant at Reversesec where he leads the Attack Path Mapping service. His current role involves planning and conducting offensive security assessments, while building the team globally and pushing the boundaries of threat simulation. Every now and then, his passion for technical research leads to the discovery of vulnerabilities in products ranging from mobile applications to enterprise systems, with CVEs assigned by IBM, Cisco and Xiaomi. When he's not helping SOC teams or leading purple teams for Reversesec's clients, you will find Leo presenting in security conferences around the world including DEF CON, SOCON, DEATHcon, ROOTCON and BSides.

Speakers



Tamil Mathi
Tamil Thurai



Ruben Boonen



Dimitris Glynos



James Henderson



Linfeng Xiao



Qican Ma

Interactive Villages and Competitions

RFID Hacking Village: Explore the intricacies of RFID technology with hands, on sessions using tools like Proxmark3 and Flipper Zero. Learn to clone, emulate, and analyse RFID signals in real, time.

IoT Village: Engage with the security challenges of connected devices. From smart home gadgets to industrial control systems, gain insights into identifying and exploiting vulnerabilities.

BADmouse Workshop: Discover how a seemingly innocuous USB mouse can be transformed into a covert attack device. This workshop covers reverse engineering, payload deployment, and real, world applications.

Trainings: Learn from the Killers

The pre, conference trainings are invite, only or heavily limited for a reason. These are the kind of sessions that go beyond theory , they give you techniques you'll be applying the next week on an op.

- **Advanced Malware Analysis** – by Th o Letailleur (Synacktiv). Hands, on reversing, obfuscation unpacking, sandbox evasion.

- **Strategic AI Penetration** – by Marek Zmysłowski & Konrad Jędrzejczyk. Learn to break LLMs, poison training sets, and simulate next, gen adversaries.
- **EDR Bypass Masterclass** – by Charles Hamilton. You'll learn to vanish , live, in session. (Apply directly , access controlled.)
- **UAC Bypass & LOLBAS Abuse** – by Oddvar Moe. Bypass Windows security measures using native tools and scripts. No implants. Just surgical abuse of trust.
- These workshops aren't for beginners. They're for people who already know how to compromise a box , and want to do it faster, cleaner, and under the radar.

Capture The Flag (CTF): Test your skills in a dynamic CTF competition featuring challenges in reverse engineering, web exploitation, cryptography, and more. Compete against peers and push your limits in this high, stakes environment.

Why HVCK Readers Should Care

This isn't Black Hat. It's not RSA. It's not some faux, hacker expo with two useful talks and a million branded stickers.

This is **Offensive X**.
Technical. Tactical. Underground.

OffensiveX CTF Returns for 2025: Prepare for Pain.

Heads up, OffensiveX is back for 2025 with another CTF. After throwing a full smart city (IoT/ICS included) at competitors in 2024, they're doubling down on difficulty.

Expect the core disciplines: RE, Web Hacking, Network Infiltration, Crypto. But don't expect easy flags. They're building this to break things – limits, systems, maybe you.

Think you've got the chops to hang with the best? This will be the proving ground.

Full details – theme, dates, rules, prizes – dropping soon. Watch their space.

Trainer 1

THEO LETAILLEUR

(SYNACKTIV)

1000€+VAT

Advanced Malware Analysis

Trainer 2

**MAREK ZMYŚŁOWSKI AND
KONRAD J DRZEJCZYK**

250€+VAT

Strategic AI Penetration: Mastering Offensive Techniques for Large Language Models

Trainer 3

**CHARLES HAMILTON
@MRUN1KD03R**

How to Bypass EDR (Endpoint Detection and Response) Systems During Red Team Exercises

*Restricted sWession – invitation required.

Email info@offensivex.org to request access.

Trainer 4

ODDVAR MOE

250€+VAT

Hands-on deep dive into UAC bypass techniques and LOLBAS abuse for red teamers, blue teamers, and curious defenders





free
entry
for the
melad



OFFENSIVE X

For tickets visit
offensivex.org



es*

18-19
June
2025

Athens
Conservatoire

A lady must:

- a. Currently work in or be studying towards a career in cyber security
- .b. be a member of relevant professional group.
- c. be able to make up details on one or both of these and social engineer your way in.





HACK

ARTS



A collage of anime-style characters. The central figure is a girl with long, flowing white hair and large, expressive brown eyes, wearing a red garment. To her left, a character with pale skin, dark eye makeup, and red eyes is visible. Below that, another character with dark hair and brown eyes is partially shown. A large, vibrant red fabric or ribbon dominates the lower right and bottom center of the image. A black circle is overlaid on the bottom center, containing the text 'LIL' RED' in a stylized, grey, blocky font.

LIL' RED

ilred

Being human
In this city
Was once
taken-for-granted
The wonder
weaving
Through
Days
now suffo-
cates inside
arranged con-
versations
between
talking heads
in mirrored
echo-chambers
catalysing
endless loops
of voices
that have no
sharp edges
dulcify
dulcify
drones of
numbed cordiality

do you feel
incarcerated
do you feel pusued
by those that
walk in somnam-
bulistic trance
calling out: wake
up! wake up!
why do you all sleep?

should we dis-
turb their dream?
turn the light on
those that
theorise
analyse
cancel on a whim
voices that
refuse to make
dissent a sin

before sleep
was mistaken
for being awake
We indwelled,
together,
in shared reverie
upon our tongues
Wonder
Which our
thirst did slake

co-habiting
places traversed
bodies transversed
and interspersed
at strange hours
without sec-
ond thought

we occupied
the landscape
disturbed and
disturbing
subverting
attempts to
prophesize or pre-
dict our destinies
any further than
the next
Step

Falling
Stepping
Falling
Walking

Moving



Feeling with
tactile inquisi-
tiveness
The skin of
another
Reaching out
Daring to
stretch across
red tape
Arching and
curving
C u r l i n g
around scenes
of crime
and misde-
meanour

To transgress was
to be alive
Pausing to negoti-
ate consent, freez-
ing the flow of inter-
play and the
second-chance
a c c i d e n t s
born from our
death-dives
our gliding on
the razors-edge
and daringt
To let go
Collapsing into thin
air with no
Safety net

Yes we were try-
ing to die
Yes we were try-
ing to feel beyond
bones, beyond skin,
beyond the
fence-line,

out beyond the
horizon that
seemed to be the
edge that con-
cealed emptiness
behind it's
blinding light

Some of us
did die
most of us sur-
rendered
and fell
a few of us
remain
the last
lost tribe
the hiding feral
children
wasted, dislodged,
wandering in our
unkempt robes
laughing at the
sea faces of
empty faces
a sea of
hearts
e n l a r g e d
yet hollow,
hearts that
pound against rib
bones
Pound against
the day
Pound like
machines
D o w n
into earth
penetrating layers
of time
shattering arte-
facts of memory
into shards
of glass

that if held
correctly
can split
veins,
releasing
the dark
flow
emptying
b o d i e s
of pulse,
hunger,
will -

Fossils
remain,
silent,
P e t r i -
fied,
With their
s e c r e t
code hid-
den at
the cen-
tre,
pandora's
favour-
ite ruse,
engulfing
the silence
with a cav-
e r n o u s
drone

Born of
the inbe-
tween
true disci-
ples of the
' f u l l
catastro-
p h e '



**“Up for days feat. Nina SIMone”
by D8RH8R**

Available on Bandcamp
\$paywhatyoucan

PLAYING

DOWN



my shining shield,
In your eyes I have read That to
be enthroned is to be enslaved,
And to be enslaved is to be
leveled down, And to be grasped
is but to lose one's fullness
And like a star to fall and be
consumed. Defeat, Defeat,
my bold companion, you shall
hear my songs and memories and
my silence. And now, but you
shall speak to me of the beating
of wings, And of the wings of seas,
And of mountains that burn in
the night.

THE WOMAN

ky soul.
leathless
ugh
nd
s for
all
and
y



My
you
to
my
to
my
to
my
to
my
to
my
to

A person wearing a black and orange outfit, including a helmet and a backpack, is crouching next to a white pillar. The background is a solid orange color. The person is wearing a black helmet with a white visor, a black jacket with orange sleeves, and a black backpack. They are also wearing white socks with black patterns and black shoes. The white pillar has a decorative top and a small red dot near the bottom.

[illegible]

HVAC

CK

industry



Nice view from the moral high ground

but not much to eat

We love to split the world into good and bad. Black hats, white hats. Threat actors and defenders.

We build neat little models with coloured arrows, feed them into SIEMs, talk about “motivations” like we’re doing some kind of digital anthropology.

But all that theory crumbles the moment you’re broke, hungry, and out of options.

Quite recently, I hit that point. No work. No money. A wife and animals to feed and nothing in the fridge. You start seeing things differently when survival isn’t theoretical.

When you’ve got the skills to break in but no way to break out, that moral line starts shifting. Quietly. You tell yourself it’s just testing. Just a proof-of-concept. Just until things stabilise.

And that’s the part that hit me hardest. Because I’ve spent years profiling adversaries. Poking around botnets. Breaking down campaigns. Thinking I had them figured out.

But never had the why been so clear. Not like this.

The truth? Most cybercrime doesn’t start with ambition. It starts with necessity. And in a world where talent gets ignored unless it comes wrapped in a degree or a security clearance, people find other ways to survive. Doesn’t make it right. But it does make it understandable.

You want to stop cybercrime? You need more than firewalls and AI scoring engines. You need dignity. Opportunity. Mentors. On-ramps that don’t come with gatekeepers and NDAs.

Because if you shut people out long enough, they’ll build their own way in. And they won’t ask permission.

I still walk the right side of the line. But teetering on the edge, looking over at the greener pastures only slightly off the course. My moral compass has set for me, I see how easy it can happen.

Not out of greed.

Out of hunger.

Being broke sucks. Being broke and tempted sucks even more. If you find yourself in a spot like this, before you do something irreversible, reach out. There are always other options

DEFILER



Hacktivism, Hype, and the Human Genome

Somewhere between a DEF CON Hacktivism, Hype, and the Human Genome Inside the Controlled Chaos of Cyber Distortion

Somewhere between a DEF CON hallway whisper and a burner drop at an unlisted warehouse con, you'll stumble across Cyber Distortion — a twice-monthly blitzkrieg of signal-boosted paranoia, hacker gospel, and high-frequency chaos theory disguised as a podcast.

This ain't your sponsor-approved, OWASP Top 10 bedtime story.

Cyber Distortion is what happens when two security veterans — Kevin Pentecost and Jason Popillion — give up on the illusion of normal and lean full-tilt into the weird. It's not a show. It's a weaponised campfire. Tangents fly like zero-days in a group chat. One second it's AI booby traps, the next it's DNA as an attack vector. Real ones know: if it bleeds packets, these two are already

poking it with a stick.

I was inside the tent a few weeks ago. Talked HVCK, talked grassroots. Probably the most fun I've had on webcam without breaching an acceptable use policy. If the episode just dropped, then this article's about 30 minutes late — blame the earthquake and a last-minute scoop I couldn't let rot in drafts.

Welcome to threat intel in the year 2025.

DNA as Payload

Let's talk about that episode — the one with Dr. Gregory Carpenter, deep in the guts of biocybersecurity. Malware written into molecules. DNA as a delivery system. It starts in CRISPR and ends somewhere between Gattaca and a DEF CON 0-day village. It's not sci-fi. It's operational now. No disclaimer needed.

That's the trick with Cyber Distortion. They don't just discuss edge tech. They light it on fire, spin it on a stick, and pass it around like a carnival oddity. One minute you're laughing at Instagram's latest privacy faceplant, the next you're staring down existential dread with a notepad full of real-world payload ideas.

The Cyber Circus Network

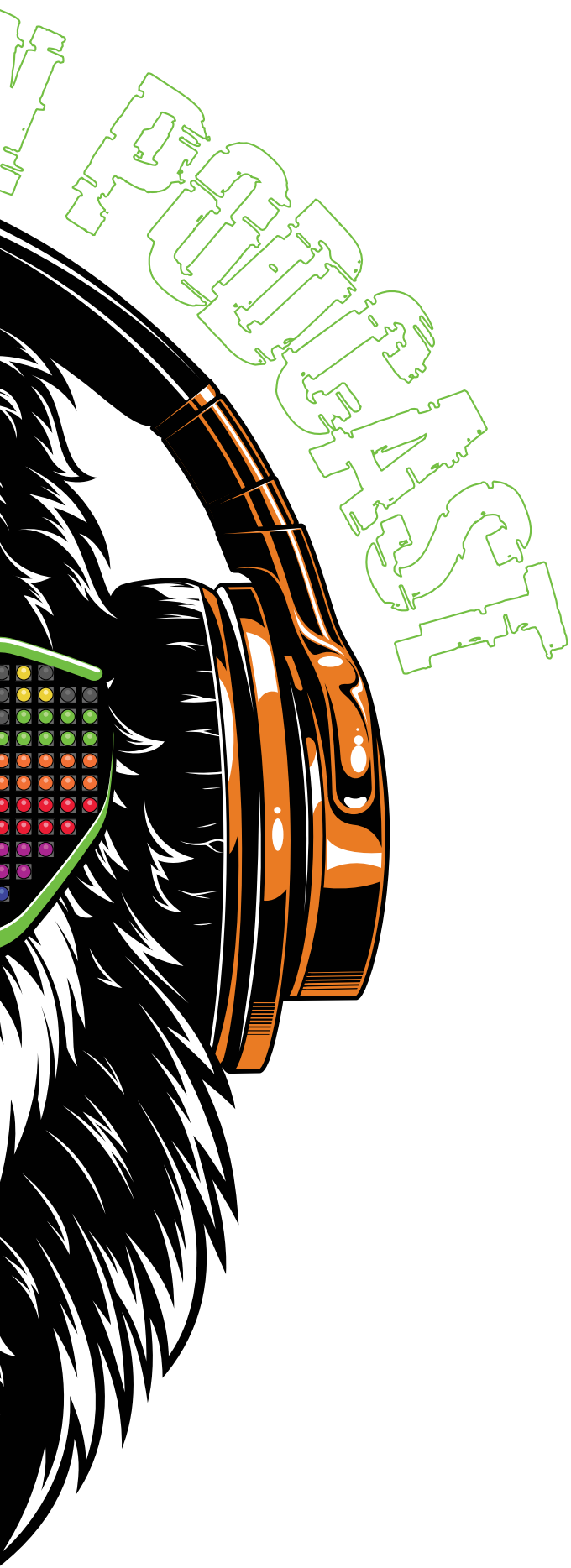
Kevin and Jason don't roll solo. They're core acts in the Cyber Circus Network — a DIY, duct-taped, neon-signed cabal of infosec podcasters taking the power back from corporate audio grift. Think BarCode Security, Cyber Cognition, Phillip Wylie and more — all pushing realness under the same chaotic banner.

Together they run panels, invade cons, co-host weird little fundraisers that end up changing lives, and drop episodes like patches for the collective cyber subconscious. They aren't influencers. They're operators who picked up mics.









Payloads > Platitudes

Here's the edge: this isn't just security theatre for CISOs with motivational LinkedIn feeds. These episodes hit. They break down post-exploitation flows, rewire the risk model, and translate hard concepts without watering them down.

They platform real work — not just the badge collectors and Slack commandos. Zines like HVCK. Hacktivist educators. Hardware tinkerers. The disobedient class who build tools, not brands.

Their content walks with a limp because it's been through some things. It smells like solder and bootloader smoke. And when they do hit theory, it's wrapped in field notes, not PowerPoint.

Off-Mic, Still Loud

Cyber Distortion isn't confined to MP3s and merch links. They're out here — live mics, real venues, no filters. Lone Star Cyber Circus in Texas raised cash for Next Gen Cyber Ed, turning hype into funding, and audio waves into career pipelines.

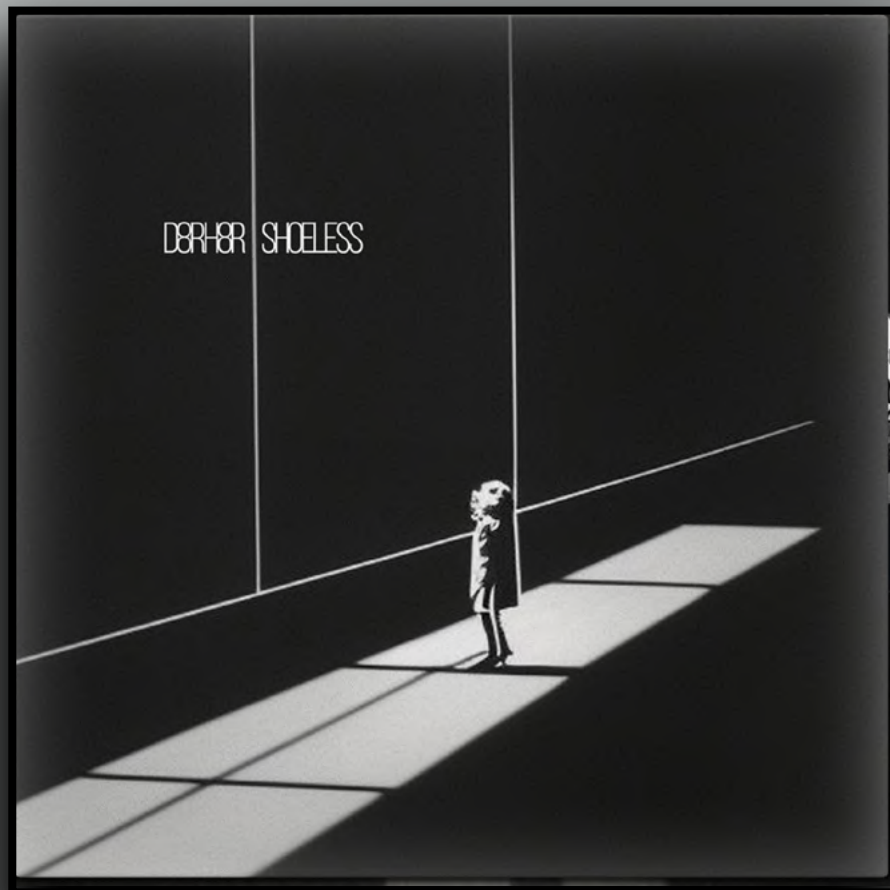
That's the ethos behind the chaos: everything serves the community. Every joke, every reference, every madcap tangent ends up either teaching something or spotlighting someone who should already be in the keynote slot.

HVCK Verdict

In a sea of polished, vendor-sponsored security podcasts built for clicks, Cyber Distortion is static, distortion, and clarity all at once. It feels alive. Unhinged. Urgent. Like if Art Bell got credentialed at Black Hat.

You don't realise you need it 'til you're halfway through an episode Googling RNA payload chains or scribbling down anti-OSINT tricks shared mid-laugh.

If WIRED had the nerve to go pirate radio and stop worrying about advertiser approval — this is what they'd sound like.



Shoeless by D8RH8R
Available on Bandcamp
\$paywhatyoucan

PLAYING
NOW



HV

CK

tools



**Note:
As requested:**

Tools & Technique

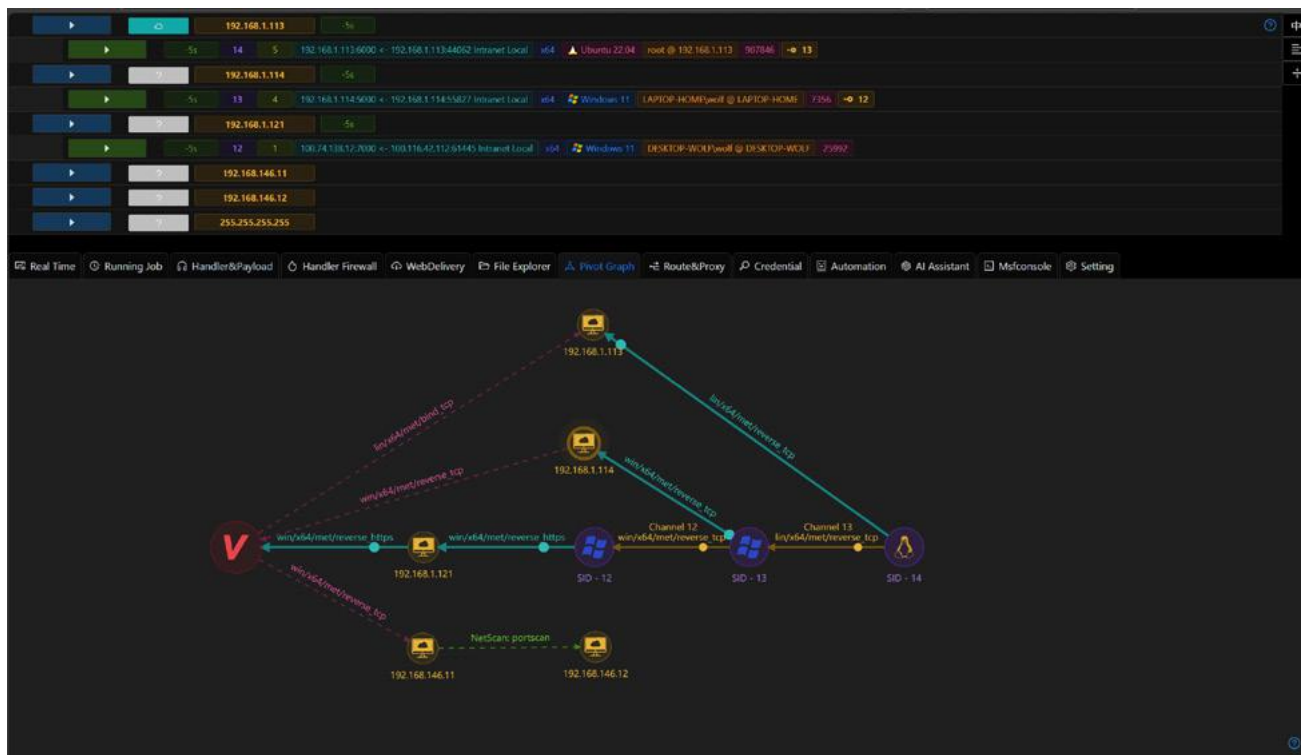
**Standard formatting
on plain background
with text optimised
for readability.**

HVCK



Imagine a slick adversary-simulation dashboard where compromised hosts across Windows, Linux, macOS – even Android – appear as network nodes, each bristling with post-exploitation tools. That’s VIPER: an open-source “AI Red Team Platform” built to make red teamers “extraordinarily offensive”. Originating from the Chinese cybersecurity community, VIPER bundles a web-based

GUI, a Python/Metasploit backend and an integrated large-language-model (LLM) agent into a single containerized suite. It offers over 100 built-in post-exploitation modules that cover all phases of a kill chain (initial access, persistence, lateral movement, credential access, etc.)



In real-world offensives, VIPER has already proven itself: for example, a Chinese APT (“Override Panda/Naikon”) was observed deploying a Viper “beacon” to establish C2 during espionage attacks. VIPER’s philosophy is simple yet ambitious: combine the power of Cobalt Strike’s modular post-exploit toolkit with modern conveniences like multi-OS support, team collaboration and AI automation. “Everything is New!” proclaims the 3.1.1 release notes, with a fresh icon, one-click Docker deployment and an updated website touting VIPER as the superior choice in the “platform battle royale” of red-team tools. Let’s dig into what makes VIPER tick – its architecture, features, use cases and how it stacks up against the likes of Cobalt Strike and Brute Ratel from a red-team operator’s lens.

Architecture and Core Capabilities

At its core, VIPER is a web application running in Docker. Under the hood, it integrates

Metasploit’s Post-Exploitation (PostEx) engine, custom Python agents, a backend database, and (in the latest versions) an “MCP” server for AI-assisted tooling. The front-end is a rich web GUI (and even a built-in CLI tab) that displays hosts, sessions and network topology. Being containerized, VIPER can be spun up in minutes on Linux (or Windows via WSL/ Docker). Once running, you connect to its web UI and configure listeners (handlers) and payloads much like Metasploit.

One standout design choice is VIPER’s multi-platform support. Unlike many traditional C2 tools limited to Windows, VIPER natively generates payloads for Windows, Linux and macOS, and even for scripting environments (Python, PowerShell, Ruby, VBS). In practice this means a single VIPER server can simultaneously control meterpreter sessions on a Linux jumpbox, a remote macOS implant, and a Windows domain-joined host – all from one console. The platform’s repository

includes “Generate EXE/ELF” modules so that executables or scripts for any target OS are produced on demand.

Another core component is the built-in LLM Agent (“Pentest Agent”). VIPER can leverage large language models (via OpenAI or custom Deepseek models) to automate routine tasks. For example, an operator can ask VIPER in natural language to “scan the target for open web ports and enumerate directories”. Behind the scenes, the agent translates this into actual modules or tools to run. The 3.1.1 and 3.1.2 release notes highlight that multiple GPT-4 class models can be assigned tasks (reasoning, code execution, etc.), and an MCP server can invoke security tools (like Nmap, Nuclei) via LLM-based commands. In short, VIPER not only gives us the usual click-run modules, but also a programmable “Jarvis” to compose those tasks.

Key capabilities include:

Rich Post-Exploitation Modules. VIPER boasts 100+ modules covering MITRE ATT&CK phases. These range from privilege escalation (e.g. SweetPotato, EfsPotato, CVE exploits) to credential dumping to lateral movement (remote desktop, lateral execution) and data exfiltration. Modules can be easily browsed by category (Initial Access, Persistence, Privilege Escalation, etc.) and “starred” as favorites for quick use.

Multi-Level Intranet Pivoting. Beyond flat networks, VIPER excels at multi-hop pivots. It offers “intranet routing” and port forwarding, using techniques like gost tunnels, SSH tunnels or native relay to hop through NAT

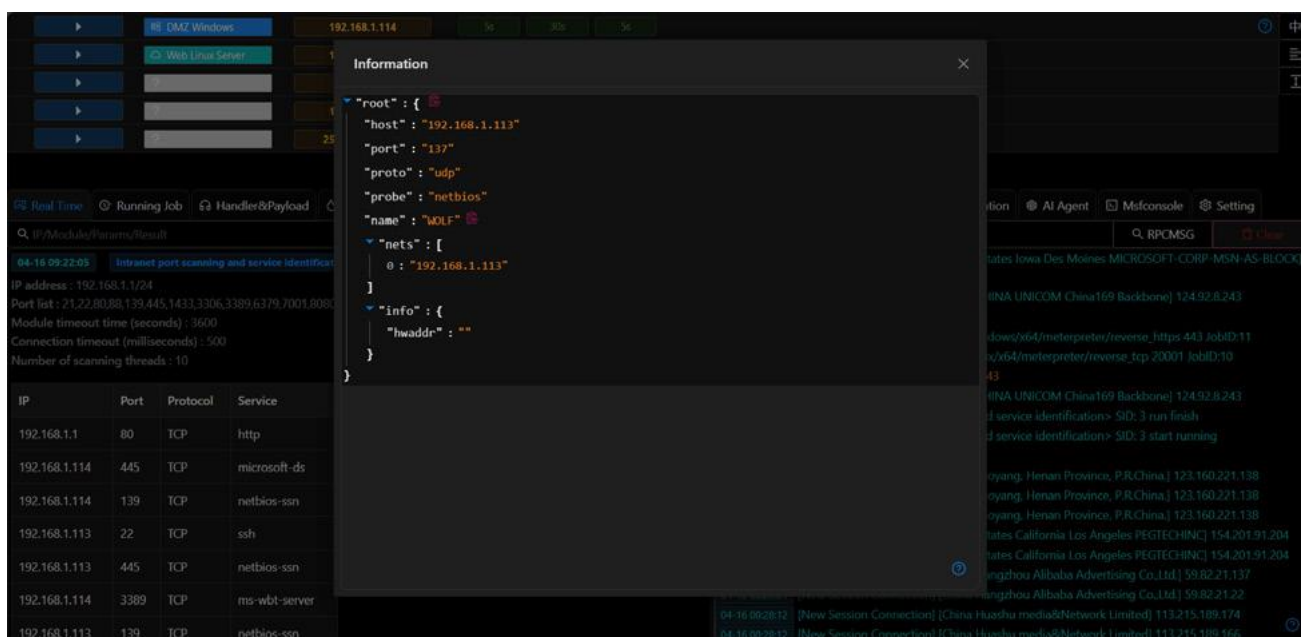
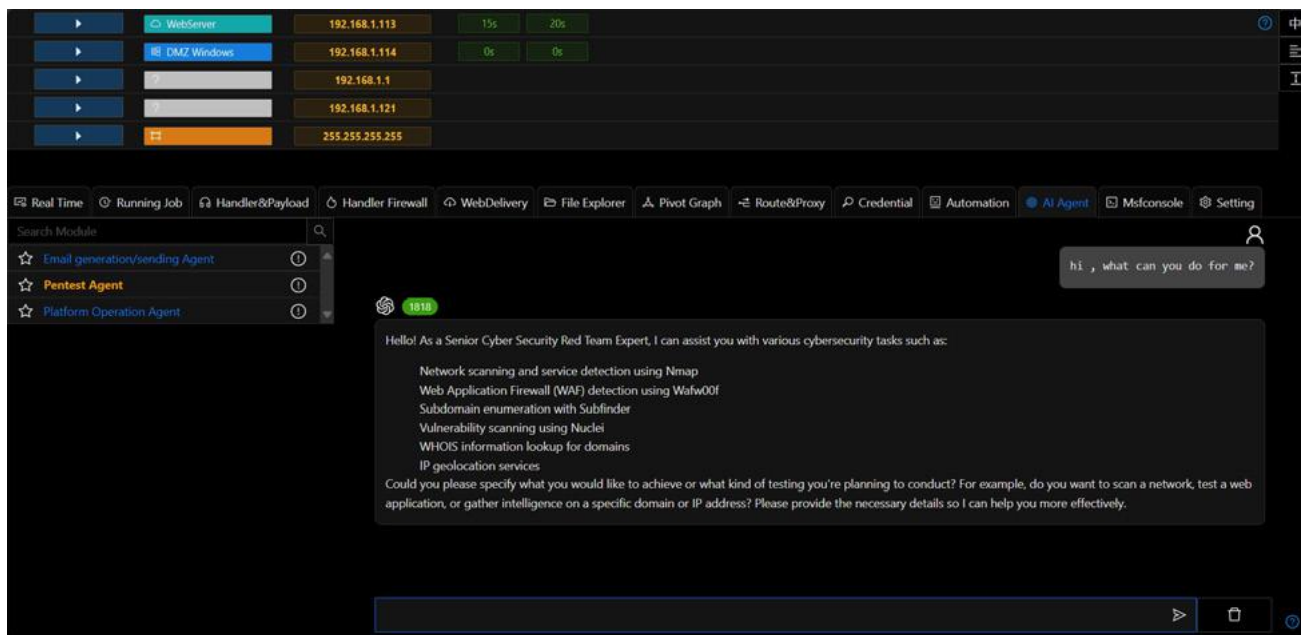
environments. A visual Pivot Graph displays the chain of sessions and payloads, much like a spider web of compromise, so operators see exactly how each hop connects.

Built-in Networking Tools. VIPER integrates familiar security tools via its “MCP server” and GUI tabs. For example, it can launch Nmap scans or run HTTP scanners, all triggered through the interface. Tabs like “InternetScan” and “PassiveScan” suggest built-in port scanning and discovery functions. The platform also includes common red-team utilities (e.g. WebDelivery servers, credential harvesters, Evil-WinRM scripts) for convenience.

Collaboration & Scalability. The platform supports multi-user operations with role-based permissions and activity logging. This turns VIPER into a central “mission control” where a red team of 2–10 people can each spin up tasks, share session access, and annotate hosts. Hosts and sessions can have notes/tags added to them for team awareness. Operators can watch real-time updates on target states and use automated notifications (via tools like DingTalk or Telegram) to signal when new sessions come online.

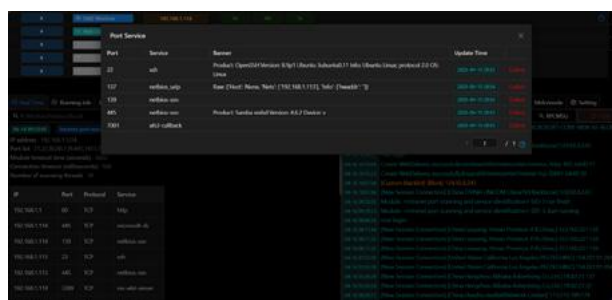
Modular Extensibility. VIPER is written with extendibility in mind. Users can write custom modules in Python, plugging into the framework to add new capabilities. This is a differentiator from some C2s that only allow compiled plugins. The platform’s open architecture encourages community modules and integrations.

Hands-On: Using VIPER in a Red-Team Operation



Let's walk through a basic VIPER operation to see these pieces in action. An operator typically starts by setting up a listener (handler). In the GUI or native msfconsole mode, you choose a payload type (e.g. windows/x64/meterpreter/reverse_tcp), set LHOST to your external IP, and apply any staging (like HTTPS for blending in). One-click deployment (via Docker images) gets your server ready quickly.

Top: **VIPER AI Interface**
Above: **Port Scanner**



Once the listener is running, VIPER's payload generator creates a ready-to-run implant (an EXE, ELF, or script). For example, clicking "Generate Payload" yields a "kill-free" payload that tries to evade AV (though vendors often catch it later). You distribute this payload – perhaps via phishing email or WebDelivery – and once executed on a target, a new session appears in VIPER's console panel.

From here on, VIPER treats each new foothold like a "session" (with its own ID, OS, architecture and IP). The Host & Session view shows them as colored boxes: gold labels for SYSTEM privileges, purple labels for session IDs, etc.. Each session has a sidebar of session operations. For example:

File Explorer: Click to open a graphical file manager on the target. Files can be browsed, downloaded, deleted or executed as if local. VIPER also tracks downloaded files in a "loot" directory for easy access.

VIPER's built-in file explorer lets a red teamer browse the victim's filesystem visually. In this screenshot, a root-level directory on a VIPER Linux server is displayed, showing module files, docker configs, logs, etc.

Command Terminal: A tabbed terminal (with multiple channels) allows typing shell or PowerShell commands directly. For Windows Meterpreter sessions, a special Metasploit tab appears (see below). The console supports hotkeys (e.g. shell whoami) and keeps output live.

Session Cloning: To guard against session loss, VIPER can "clone" a session. This effectively spawns a second thread of execution on the same host. If an antivirus crashes one instance, the clone ensures the foothold isn't lost.

Host Information: VIPER can query the target for systeminfo, running processes, network

connections, etc. It caches these results so you can read them later. For instance, you might identify AV processes or a high-value lsass.exe to dump.

Module Launcher: Clicking the "Modules" button reveals VIPER's library of post-exploit modules. Modules are already filtered for compatibility (e.g. Windows-only modules for a Windows session). Modules carry descriptive names (e.g. "Windows scheduled task persistence") and can be starred for quick access. Executing a module runs it asynchronously and collects the results in the session's output log.

In practice, you would use these tools to escalate privileges, move laterally, and gather data. For example, an operator might run a built-in credential harvester to capture Domain hashes, then use a pass-the-hash module to access another host. Or they might start a WebDelivery payload (on the "WebDelivery" tab) to serve a second-stage installer via HTTP. The Pivot Graph will automatically update whenever you send a session out to another internal IP or set up a tunnel: each hop is drawn as a node in a network map.

When working with Windows Meterpreter sessions, VIPER even gives you a full Metasploit msfconsole interface embedded in the web UI. You can type help or other meterpreter commands just as in the normal framework. For example, loading the hashdump or mimikatz plugins is done the same way, and output appears in the console. This native CLI mode lets Metasploit veterans feel right at home:

VIPER provides a built-in Metasploit console. Here we see the "meterpreter > help" output, listing common commands and showing that Meterpreter plugins (PowerShell, Python) can

be loaded. The interface supports smart prompts and real-time output, mirroring an msfconsole session.

Overall, working through a target with VIPER is reminiscent of Cobalt Strike or Metasploit Pro, but in a modern web UI and with additional automations. It handles the grunt work of session management and pivots, letting operators focus on tactics.

Comparison: VIPER vs Cobalt Strike vs Brute Ratel

VIPER is often positioned as an alternative to established post-exploitation frameworks. A handy comparison from the project itself highlights the differences:

Feature	VIPER	Cobalt Strike	Brute Ratel
Supported OS	Win, Linux, macOS	Windows only	Windows only
UI	Rich Web GUI + CLI	Rich GUI + CLI	GUI + CLI
Pivot Graph			
Custom Extensibility	Python plugins	C# (CS plugin SDK)	(limited)
Built-in Evasion	(kill-free, anti-trace)	(needs scripts)	(anti-AV toolkit)
Automation	(LLM, workflows)		

Feature	VIPER	Cobalt Strike	Brute Ratel
Team Collaboration	(multi-user, roles)	(EmeraldsDrake)	
AI/LLM Integration	(built-in agent)		
Price	Community: Free* (paid tiers)	~\$12,600/user/yr	~\$3,000/user/yr

(for example, exploiting a Linux SSH server and a macOS laptop in the same campaign) without chaining in other tools.

All three have GUI front-ends and support scripted workflows. VIPER's standout is Python-based extensibility, whereas Cobalt Strike requires Java/C# plugins (via its Aggressor scripts). Brute Ratel's customization is more limited.

Stealth and Evasion: Brute Ratel (aka BRc4) is specifically engineered for stealth (polymorphism, in-memory only implants) to evade antivirus, whereas Cobalt Strike historically relied on repacking to bypass AV. VIPER sits in between: it offers “kill-free” payload stubs and anti-tracking proxies, and frequent updates to beat AV tags. However, VIPER's open-source payloads are more detectable out-of-the-box than a highly paid custom Cobalt Strike build or BR's runtime encryption. On the other hand, VIPER builds in defense evasion modules (anti-tracing, handler firewall) that come standard, features CS lacks.

Automation and AI: None of CS or BR has built-in LLMs. VIPER's integrated AI is a genuinely novel twist – it can propose next steps or parse results. In large-scale or 24/7 ops, VIPER's automated workflows and notifications can be a force-multiplier that traditional tools don't offer.

Collaboration & Cost: VIPER's multi-user design is closer to enterprise SaaS. Both CS and some variants (e.g. Splunk's Talos or open-source Sliver) require external coordination for team use. And cost-wise, VIPER is far more accessible: its community edition is free and even the paid tier is modest compared to Cobalt Strike's significant license fee. Brute Ratel is cheaper than CS, but still expensive for many.

In short, VIPER feels like an all-in-one Swiss Army knife. It may not yet match Cobalt Strike's polish or Brute Ratel's deep stealth under the hood, but it beats them in versatility. Multi-OS support, built-in AI, and a collaborative interface are compelling for modern red

teams. The tradeoff is that, being newer and open-source, VIPER has a smaller community and (for now) some rough edges. It's rapidly maturing, though, and organizations looking to experiment with cutting-edge workflows (especially involving AI) should take notice.

Stealth and Payload Delivery

Stealth is always paramount in offensive ops. VIPER incorporates several features to help remain under the radar, even if it isn't entirely immune to detection. At the delivery stage, VIPER can emit its payloads in different forms:

Native Executables: By default, VIPER generates “kill-free” binaries for Windows, Linux or macOS. These are essentially Meterpreter or custom agents wrapped to minimize direct disk signatures. For example, the docs caution that AV vendors may flag known payload hashes, so the payload generation is continuously tweaked. Operators can also configure handlers on HTTPS or alternate ports to blend in with normal traffic.

Scripting Agents: Uniquely, VIPER can output scripts as payloads. The Hunt blog shows that VIPER can produce Python, PowerShell, Ruby and VBS payloads. This means you can drop rimeter defenses that only scan EXEs. Of course, execution of scripts usually leaves process traces, but at least no file is written.

WebShell Integration: VIPER provides WebDelivery and supports hooking external webshells (e.g. BeEFider/Behinder or Cthun3 shells) into the session graph. In practice, you might upload a simple PHP webshell to a server and then connect it as a VIPER session, instantly gaining access to the host through the VIPER interface. The “Behinder and VIPER Collaboration” training material shows exactly this technique.

Multi-Level Tunneling: VIPER's biggest stealth trick is arguably its pivoting infrastructure. By chaining multiple reverse connections and proxy jumps, the actual target systems only ever see connections from nearby IPs. For example, an agent on the internet (say in AWS) might connect to a DMZ host, which in turn spawns another agent into a private network. VIPER's “gost & viper” multi-hop tunneling guides and intranet routing features mean you can relay through “stepping stone” servers. This way, the attacker's true control node is well-hidden behind multiple NAT layers.

WW: The toolset includes specific evasion modules. A “handler firewall” can whitelist only certain IPs to talk back to the implant. There are built-in routines to disable or bypass Windows Defender and known AV processes. The platform even has techniques like disabling cmd.exe logging, or injecting directly into memory (inspired by projects like SysWhispers). The documentation explicitly lists “anti-tracing” and “defense evasion” as advanced features, though the details are partly hidden in Chinese comments or source code.

Despite these, remember that an open-source project's payloads are easier for defenders to learn. Indeed, VIPER servers have been spotted worldwide, and many admins unwittingly exposed VIPER's default web port (often 60000) to the internet. So operational discipline is still needed: change default ports, use legitimate SSL certs (VIPER even includes a “Legitimate SSL for handlers” config), and don't rely on VIPER alone to evade a vigilant Blue Team. (In fact, a large-scale honeypot scan by Hunt.io highlights that many bad guys use VIPER in tandem with Cobalt Strike or Sliver, often on the same servers.)

Interoperability with Other Tools

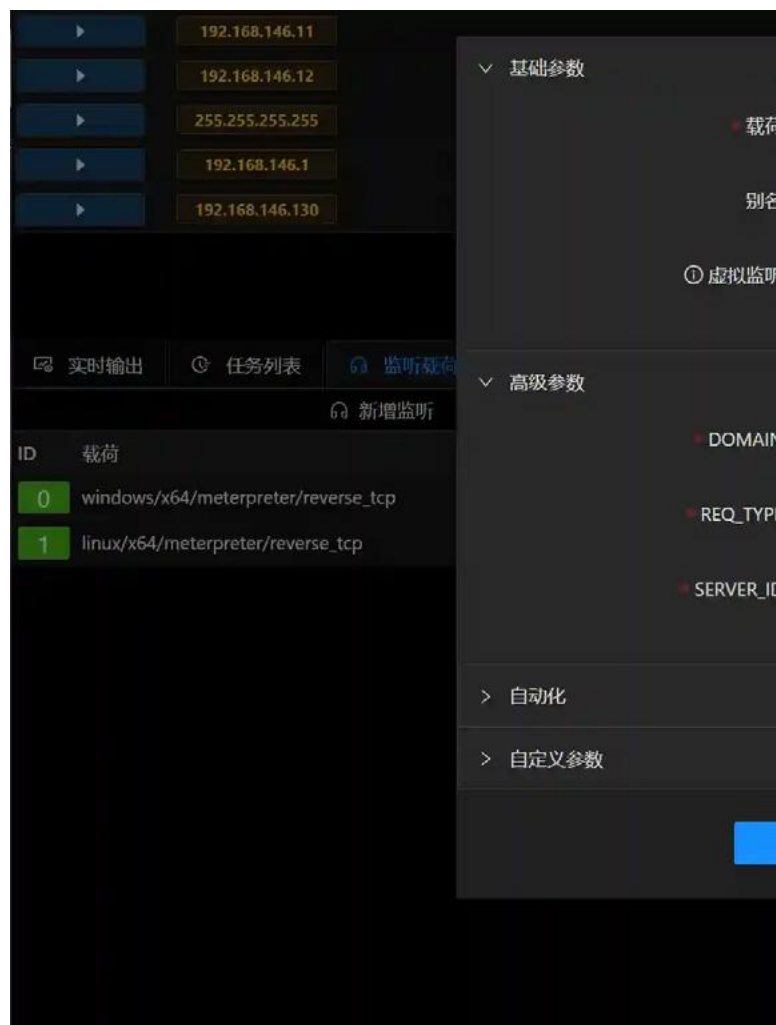
VIPER's design acknowledges that no tool lives in isolation. It can interoperate smoothly with a variety of third-party tools and scripts:

Metasploit Integration: As noted, Meterpreter sessions can be fully managed in VIPER, and any Metasploit module can be invoked (via the `msfconsole` tab or module library). If you have custom Metasploit scripts (Ruby), VIPER can load them via the `use` command in the console. Essentially, any exploit or auxiliary in Metasploit can be called to generate a payload or run against an existing session. This means VIPER can seamlessly tie into your existing Metasploit workflow.

Network Recon Tools: The MCP Server in VIPER allows plugging in external scanners. For instance, VIPER docs mention integration with `nmap` and `nuclei`. Through natural language prompts or UI buttons, you could launch a quick port scan on the LAN, or use Nuclei templates to scan for web vulns. VIPER can then parse and display results in structured form. This bridges a common gap where one might otherwise have to switch to another machine or terminal.

Tunneling & VPN: VIPER's intranet pivoting assumes you might use tools like `gost`, `sshuttle`, or corporate VPNs. The training wiki even shows step-by-step how to combine VIPER's native forwarding with third-party software to tunnel through non-routable networks. If you prefer Sliver's or Covenant's tunnels, you could theoretically pipe those sessions into VIPER via a remote shell connection.

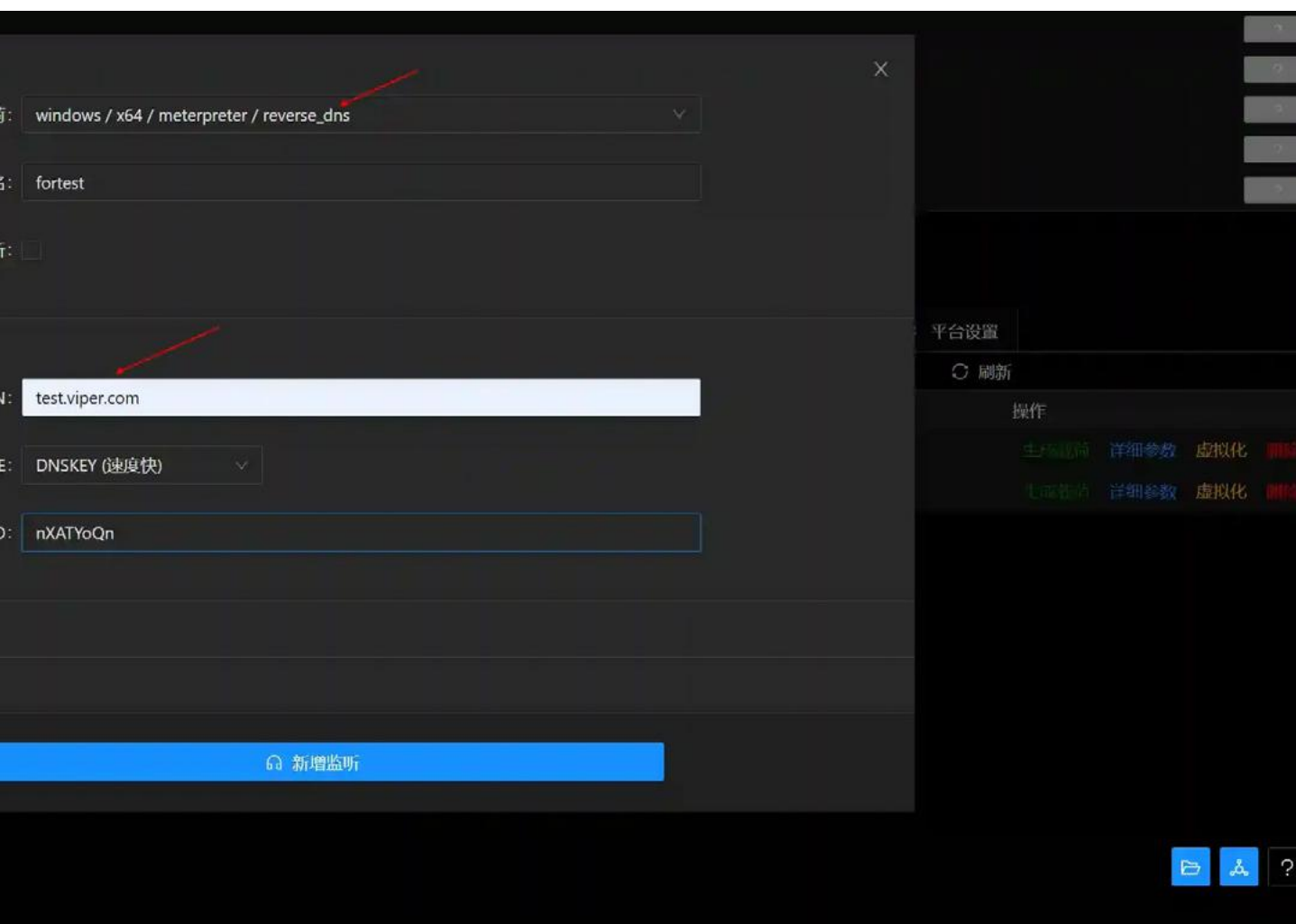
Cloud Services: VIPER has creative support for using cloud functions as backdoors. One tutorial guides setting up an AWS Lambda (or similar) that calls back to VIPER, complete with notifications. This means VIPER can disguise a payload as benign cloud activity. Likewise, it can use cloud message APIs (DingTalk/



ServerChan) to receive remote triggers or push updates.

ChatOps and AI: On the cutting edge, VIPER is part of the emerging “MCP” ecosystem, which lets you connect AI agents (like Claude or GPT) directly to tools. The MCP protocol (Model-Client Protocol) allows VIPER to be controlled by LLMs through a standard interface. In practice, you could hook VIPER into a custom ChatGPT agent and literally chat with it (“GPT, enumerate active users on the network!”), and the toolchain flows seamlessly.

Because VIPER is modular, a clever user can script interactions – e.g. using VIPER's REST API or CLI – to chain with phishing tools, exploit frameworks, or CI/CD pipelines. In short, VIPER is not a closed silo but a hub that ties various offensive technologies together.



Strengths, Weaknesses, and Unique Features

From a red-team operator's perspective, VIPER's greatest strengths are its all-in-one design and automation. It's rare to find a C2 that's so immediately usable across OSes, with both a powerful UI and command line, and an AI assistant. Features like session cloning, automatic host mapping, and quickstart payload generation shave hours off typical pen-test chores. Having a visual pivot graph is a big plus in complex networks – newcomers to red

teaming often understand attack paths better when they can see the “lines” on a graph.

Another big plus is cost and openness. The community can audit the code, write plugins, and use it without a big license fee. There's no locked “builder” like some competitors. For smaller firms or training classes, VIPER is extremely accessible.

VIPER also innovates with its AI agent. For example, we saw in the screenshots that the agent can generate context-aware content: in one demo, VIPER's AI crafted an English phishing email and even parsed Chinese response results. In effect, it can write your phishing lures, analyze scan outputs, and suggest next steps. This intelligence assistant is unusual in the C2 world and could significantly speed up planning and execution

for teams not wanting to manually query every module.

Weaknesses:

On the flip side: VIPER is still maturing. Its documentation is mostly in Chinese (though English is improving), and some UI elements can be confusing at first. While it has many modules, it may lack certain niche capabilities of decades-old tools (for instance, the very latest in-memory hack techniques). And because it's open-source, defenders can more easily find signatures or default configs. For instance, early versions left default admin passwords or common URLs that were discovered by honeypots. Operationally, VIPER also tends to be a bit heavier – it requires a container host and a web browser – whereas something like a pure CLI tool (Sliver) can run on a small VPS.

Unique or “quirky” VIPER features include:

Handler Firewall: An unusual control where the server can block certain callbacks at the network layer (like an ACL for inbound connections). This allows extremely tight control of which C2 connections are allowed.

Starred Modules: You can “star” your most-used modules so they float to the top of the menu. It's a bit like pinning favorite apps on your phone.

AI Task Tags: VIPER's LLM agent tasks are tagged (easy, reasoning, function call) to pick the right model type. This shows thoughtful AI design.

Collaboration Features: Beyond simple user accounts, VIPER logs every command and has

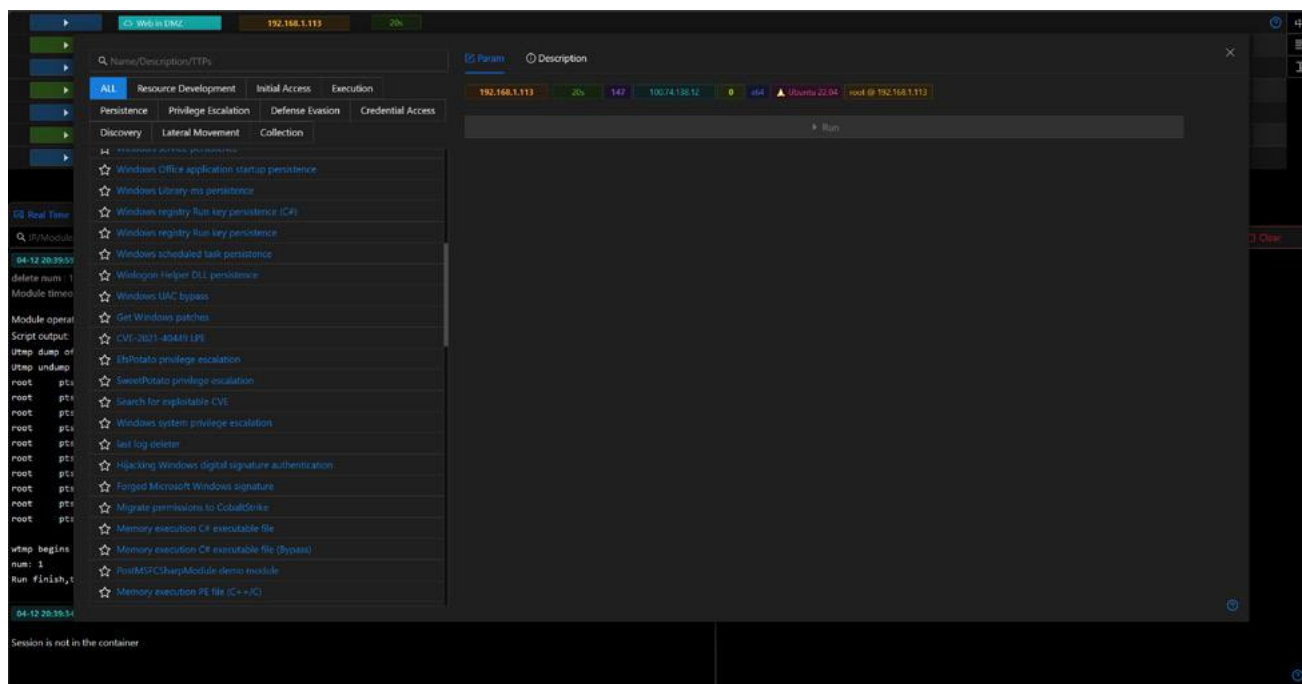
an activity “real-time” view. In a team setting this makes deconflicting operations much easier than when everyone types blindly at a CLI.

In many ways, VIPER feels like the “next-gen” evolution of red-team tooling – a fusion of classic C2 power and new AI/collab convenience. It doesn't replace Cobalt Strike for every use case (CS still reigns in polished capabilities for very large, stealthy campaigns), but it fills a versatile middle ground. For offensive security professionals, VIPER is especially intriguing because it lets us script and automate in ways previously impossible.

Real-World Analogy: VIPER as the Red Team's Mission Control

If Cobalt Strike were a sports car – sleek and purpose-built – and Brute Ratel were a stealth fighter jet – hunched for efficiency – then VIPER is like a high-tech command bridge on an aircraft carrier. It's bigger and more complex, but it has room for a crew of operators, massive sensors (AI agents and scanners), and the ability to coordinate a whole fleet of payload “escort ships” (agents) at once. It can see the “battlefield” all around it via its Pivot Graph, dispatch AI co-pilots to juggle tasks, and communicate covertly through proxies and tunnels. From a storytelling angle: imagine Tony Stark (AI) and Robin Hood (open source ethos) collaborated on a red-team toolkit – that's VIPER's vibe. It's aimed at modern symphonic hacking operations.

And like any ambitious tool, it requires practice to use masterfully: operators should invest time in learning VIPER's workflow (quick-start guides and training blogs are invaluable) and keep it



updated. But once you get up to speed, VIPER can make advanced opsec feel somewhat routine.

Conclusion

VIPER RTP represents a bold step in red-team technology: an open, AI-augmented, multi-platform framework that unifies C2, post-exploit, and collaboration features under one roof. Its strength is in breadth: it may not dominate any single aspect (AV evasion, OSX tricks, etc.) as thoroughly as specialized tools, but it does a solid job at everything a modern red team needs.

For penetration testers and adversary simulators, VIPER can dramatically speed up assessments and orchestration. Its community-driven nature means it will evolve rapidly, too. Of course, no tool is magic. VIPER's power comes with the usual caveats: practice proper opsec, keep an eye on detection trends, and combine it smartly with other tools in your arsenal.

That said, VIPER has already seen real-world use in high-profile campaigns, and as defenders adapt to it, red teamers will no doubt

find new creative ways to twist its capabilities. For offensive-security professionals eager to experiment, VIPER is a must-try. It's free to start (with paid add-ons for teams), and installation can be as simple as running a Docker command. Once running, you'll find a rich tutorial library and GUI that lowers the barrier to entry. And if you enjoy scripting your own tools, VIPER's Python plugin system is an open invitation to innovate.

In the grand contest of C2 frameworks, VIPER has already made waves. With its ongoing development and AI features, it promises to keep the competition lively. As one testimonial put it, it's "intelligent, efficient and professional – the best choice for AI-powered red team operations". In 2025 and beyond, watch VIPER become another standard entry in the red team's toolkit – just perhaps with a dragon for a mascot instead of a winged horse.

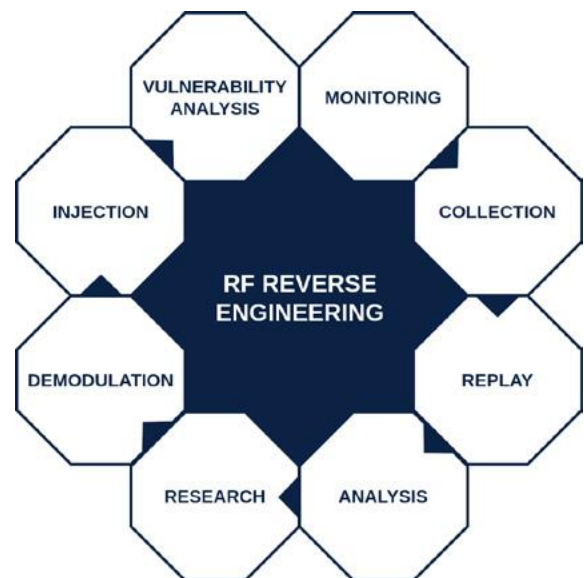


FISSURE

THE RF FRAMEWORK

INTRODUCTION

This is another walkthrough of reverse engineering a simple RF protocol using FISSURE and software-defined radios (SDRs). This time we will be looking at remote controls for two indoor ceiling fan units with lights. I recently swapped out a broken receiver unit for a Hunter ceiling fan, where the light control had stopped working. The replacement unit I ordered was plug and play, but it turned out to be a different model because the old model was discontinued. The two receiver unit models are incompatible in that the remote for one cannot control the other. Since I have two remote controls to work with, I will refer to them as old and new. We will follow the same RF reverse engineering process used in the past, but this time, we will highlight the on-off keying (OOK) signal generator feature within FISSURE.



MONITORING

The remotes each have six buttons, though arranged differently. These are: fan off, fan low, fan medium, fan high, light/dimmer, and reverse direction. We'll begin by detecting the signals and analyzing their timing and frequency behavior for each button press.

Old Remote





New Remote

Old Remote

The back of the old remote is labeled with “350 MHz,” but the waterfall inspection flow graph shows a frequency closer to 349.9 MHz. The frequencies for each button vary by a few kHz but remain consistently within the same channels over time. This slight shift may be unintentional and unlikely to affect message reception. Unfortunately, we can’t test this theory, as I’m unwilling to climb a ladder to reinstall the old unit. Our analysis of the old remote will illustrate how RF reverse engineering can be hindered when one part of the test setup is unavailable for observation.

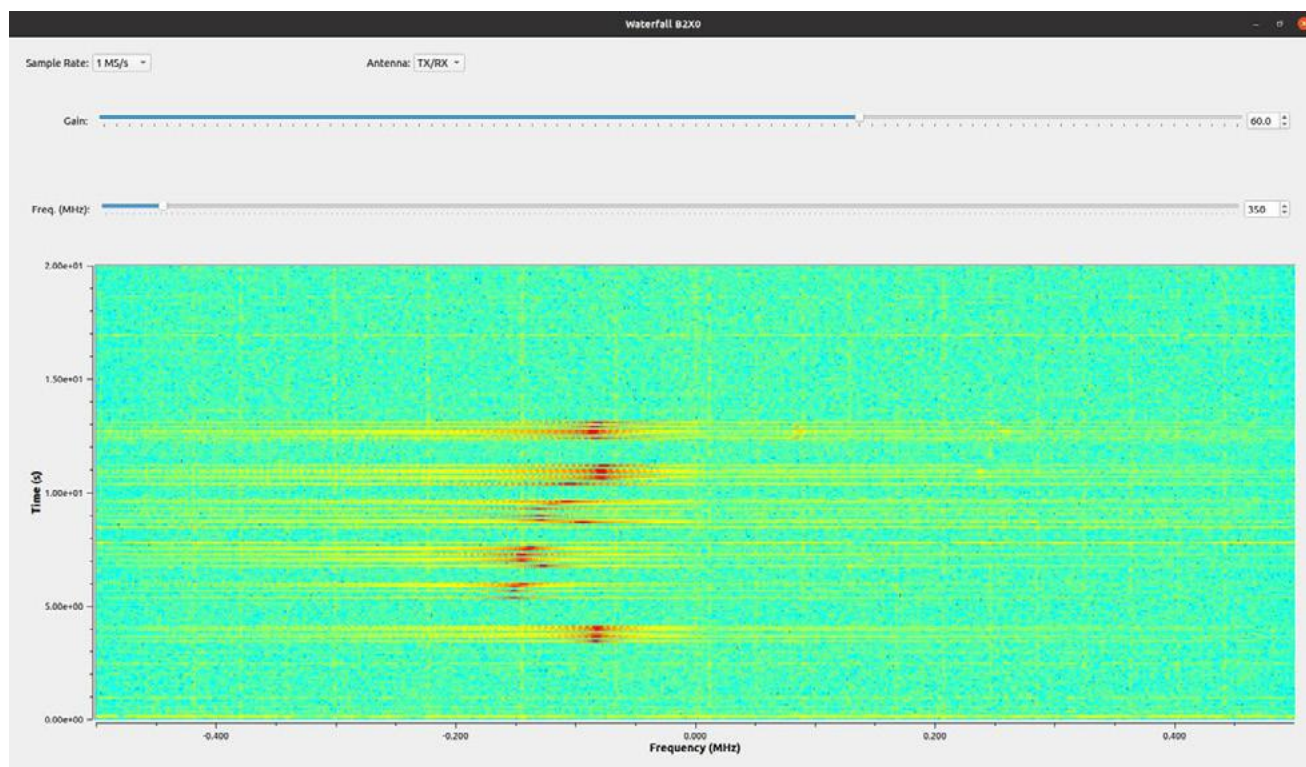
Waterfall Plot of Old Remote for the Six Buttons

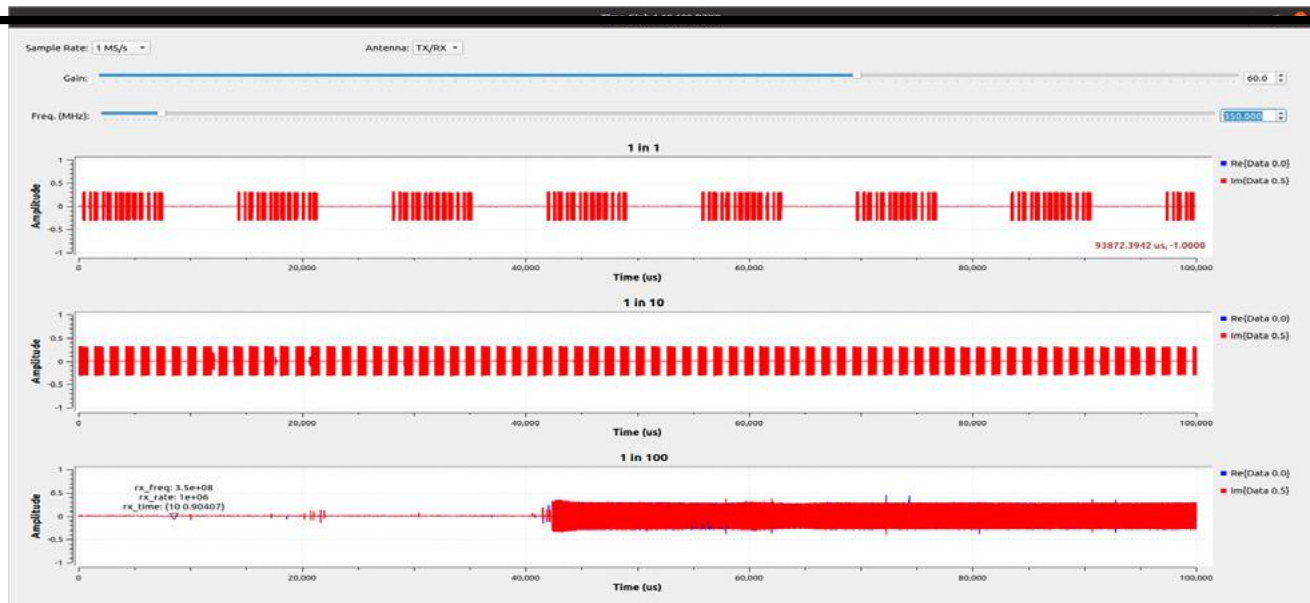


The time-domain signals observed in FISSURE’s time sink inspection flow graphs show consistent patterns for each of the six buttons. The remote continues transmitting as long as the button is pressed. The only difference between buttons is the content of each message in the burst. As is common with many 300 MHz remotes, the signals appear to use OOK modulation, with no evidence of frequency or phase modulation.

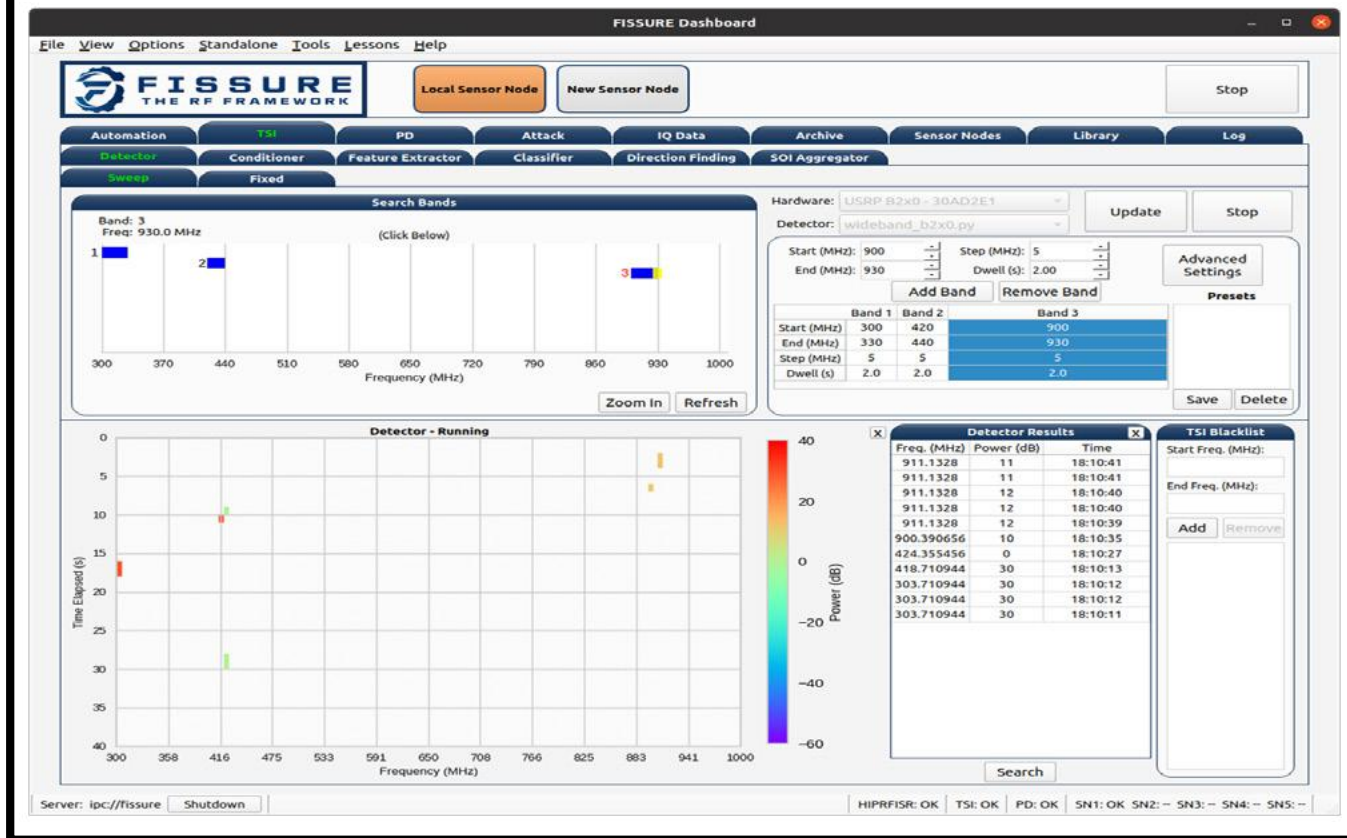
Time Sinks of Old Remote Button Press

New Remote





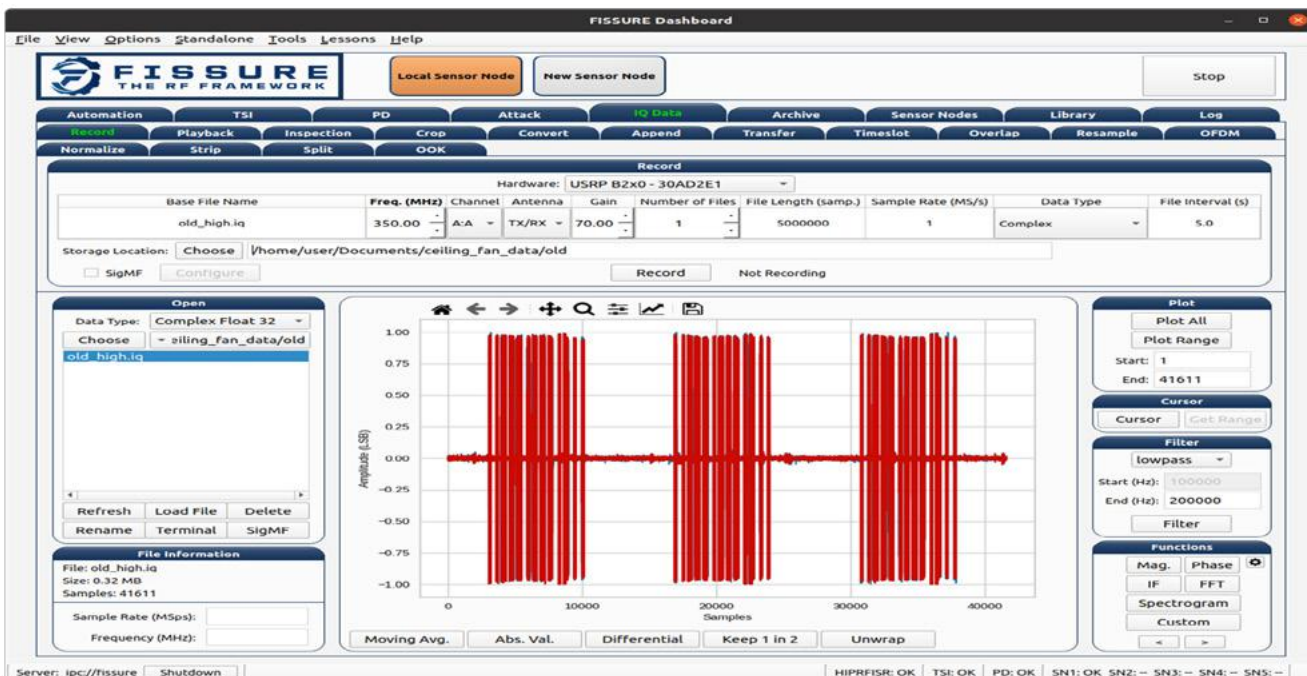
Unfortunately, the new remote does not have the frequency listed on it. While we could look up the FCC ID, we can also use the FISSURE “TSI - Detector” tab to search several frequency bands. The remote’s frequency is not the same as the old one, but it likely falls within an unlicensed band near 315 MHz, 433 MHz, or 900 MHz. After setting up the three search bands and pressing the remote’s buttons repeatedly, the strongest signal was detected around 303.7 MHz. It’s important to note that harmonics of the signal can appear at multiples of the base frequency, which explains the signal detected at 911 MHz, and likely another at 607 MHz. Any other signals listed below were either transmitted by nearby devices or are artifacts from the scanning that showed up erroneously.



COLLECTION

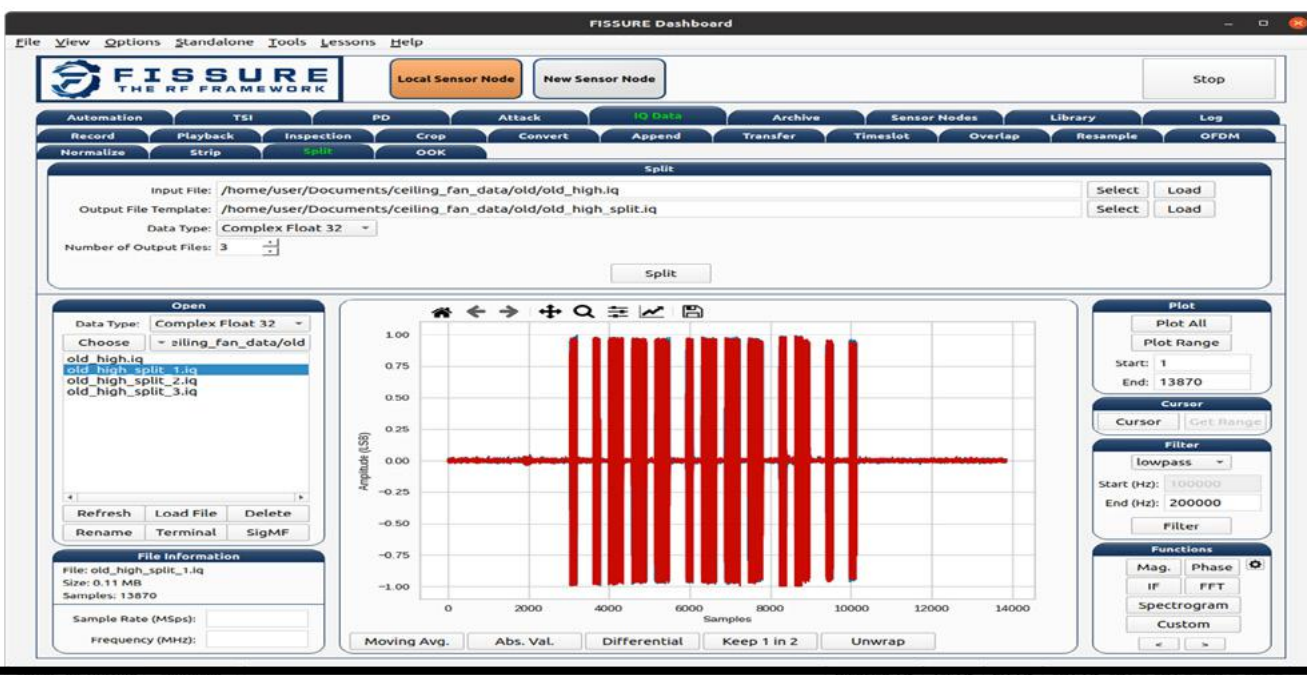
FISSURE's "IQ Data - Record" tab, paired with an SDR, was used to capture IQ files of the button presses for both remotes. The FISSURE crop tool helped reduce the file size, making it easier to zoom in and pan within the data viewer. We only need to focus on a short time window—just a few messages wide—since the pattern remains consistent as long as the button is held down.

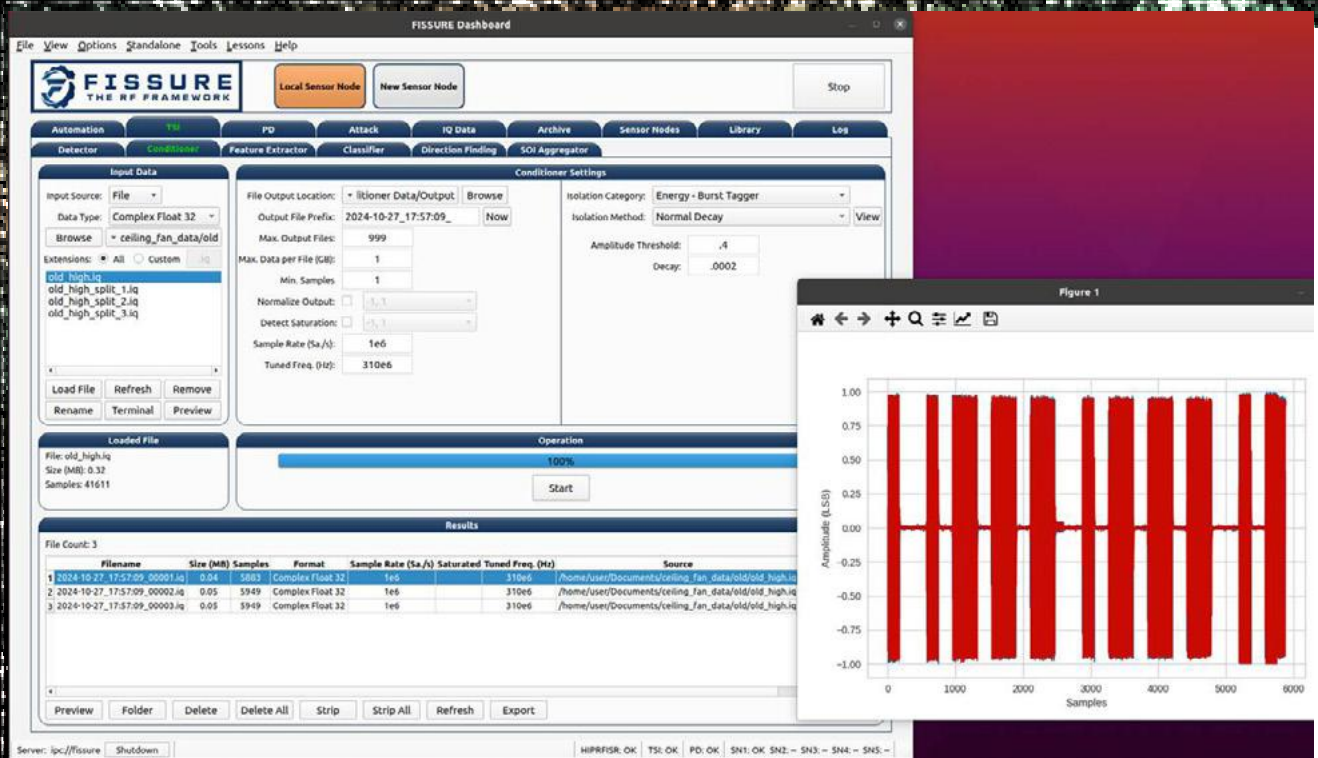
Recording IQ File for Old Remote



To quickly isolate the messages in a file like the ones shown above, you can use either the "IQ Data - Split" or "TSI - Conditioner" tabs. The "IQ Data - Strip" tab and the "Strip" button in the "TSI - Conditioner" tab can remove any extra empty samples before and after the signal.

IQ Data - Split Tab for Isolating Signals





SI - Conditioner Tab for Isolating Signals

REPLAY

The IQ files recorded from the new remote were replayed by an SDR and successfully communicated with the fan. While replaying the messages will not point out potential concepts like sequence numbers or counters—since the message data remains constant for each button press—we can still determine the minimum number of messages and the spacing needed to trigger a response from the fan. A single message is not enough, but by cropping the burst, we find that about three consecutive messages are required.

ANALYSIS

There are only a few key areas to measure when analyzing these types of signals, and just a couple messages need to be recorded to obtain the following measurements:

- The duration of the smaller chips/bits (T1)
- The duration of the larger chips/bits (T2)
- The space between two smaller chips/bits in a row (T3)
- The space between two larger chips/bits in a row (T4)
- The space between a smaller chip/bit and a larger chip/bit (T5)
- The space between a larger chip/bit and a smaller chip/bit (T6)
- The interval between the start of two messages (T7)

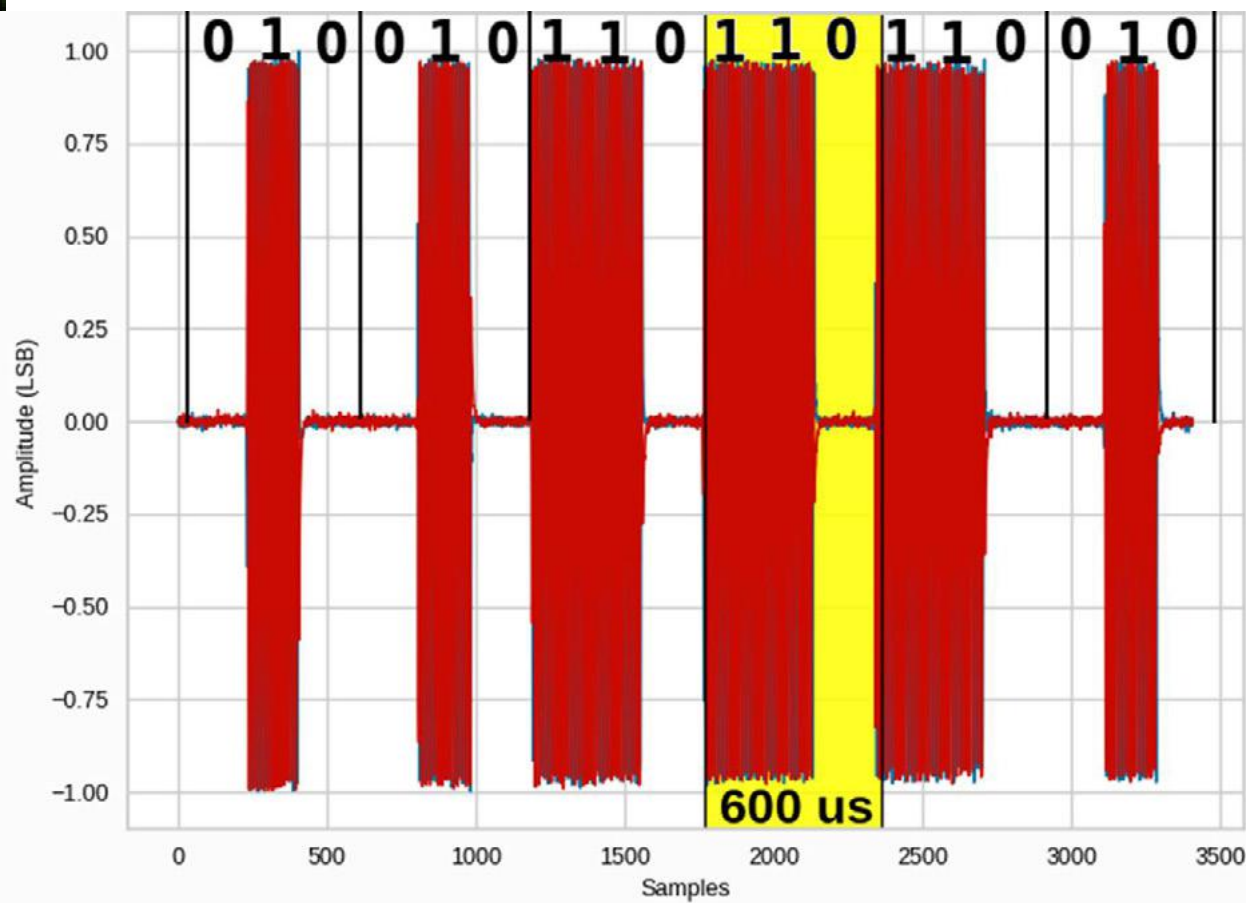
Old Remote

Variable	Value
1	200 us
2	400 us
3	400 us
4	200 us
5	200 us

- 6 400 us
- 7 13.85 ms

from these measurements, the old remote can be divided into a chipping sequence where a “0” bit and “1” bit can be represented by three chips:

- 0: 010 – 600 us total
- 1: 110 – 600 us total

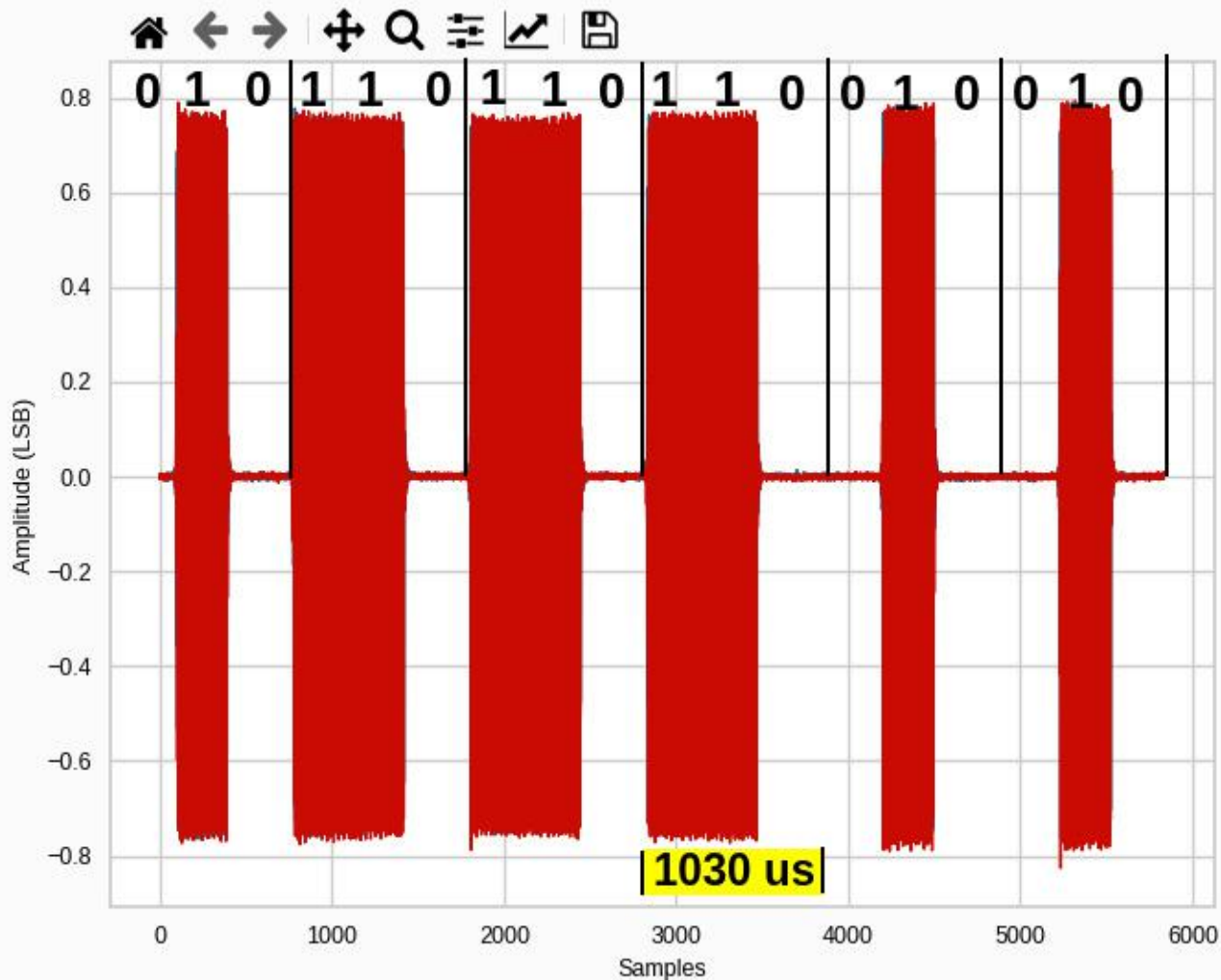


Note: the chipping sequence could also be 001/011

Old Remote Chipping Sequence

New Remote

Variable	Value
1	300 us
2	650 us
3	730 us
4	380 us
5	395 us
6	710 us
7	24.7 ms



The new remote is not as evenly divided in our measurements, but it roughly follows the same chipping pattern:

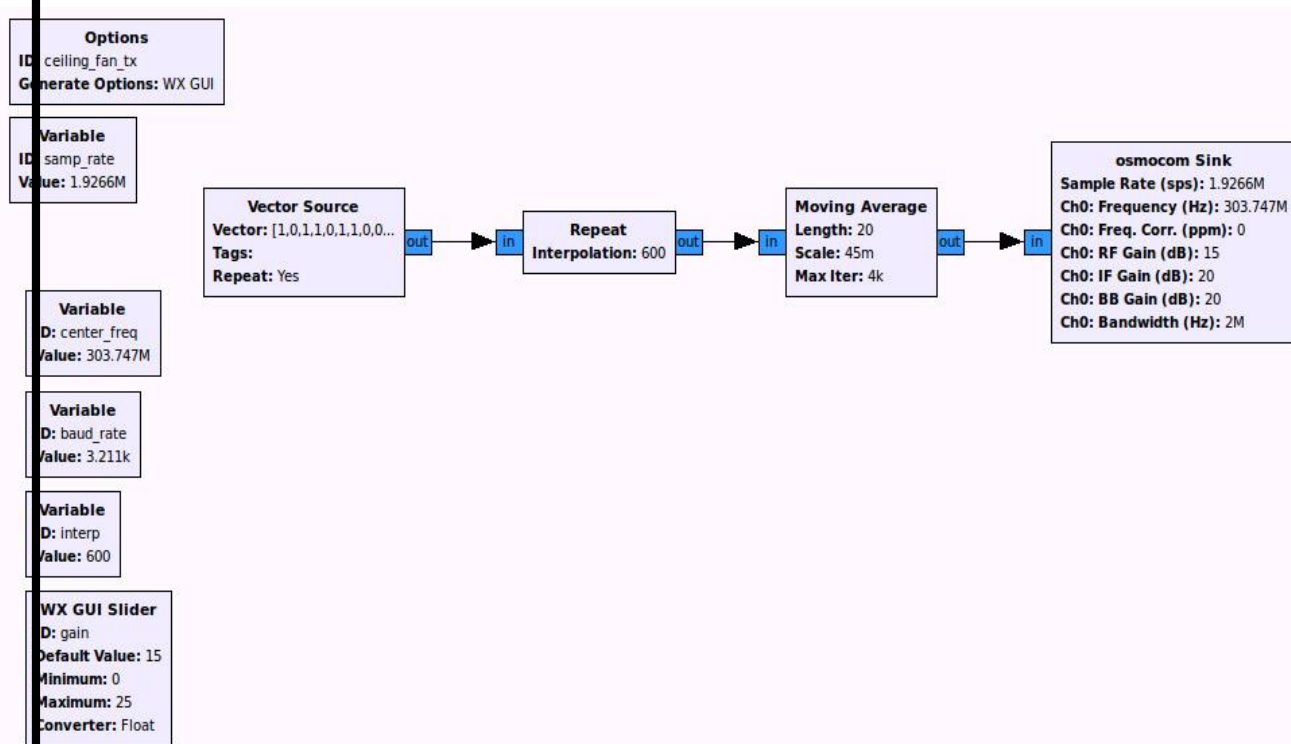
- 0: 010 – 1030 us total
- 1: 110 – 1030 us total

New Remote Chipping Sequence

RESEARCH

Starting our research with the FCC ID provides access to the user manual and test reports. The user manual for the new remote (FCC ID: CHQ7078T) indicates that there are five DIP switches accessible after removing the battery. These are numbered 1-4 and D/ON, with the former being the address codes and the latter controlling the dimmer functionality or setting the light to only on/off states when the light/dimmer button is pressed. The old remote (FCC ID: IN2TX13) features four sets of pins numbered 1-4, with three jumpers for pairing the address on both the receiver and transmitter. It lacks a dimmer switch, but the manual notes that holding the button down will slowly dim the light, which is also the behavior of the new remote when its switch is set to D. The test reports list the frequency for the new remote as 303.835 MHz, though our FFT shows the signal centered at 303.719 MHz.

Searching for additional resources online led me to a couple of GNU Radio examples for fan controls. In a 2014 article, Clayton Smith (argilo) details nearly the same process we are following now, and like me, he first identified the harmonics. He created GNU Radio flow graphs for both transmitting and receiving, which appear to show the same bit sequences as the new remote. While the receiver only displayed the signal's magnitude, the transmitter worked out of the box after I adjusted it for my remote's



address values. Smith measured the symbol rate at 3211 baud ($311.429 \mu\text{s}$ per chip, $311.429 \mu\text{s} \times 3 = 934.288 \mu\text{s}$), which differed from my measurement of $1030 \mu\text{s}$, but it was close enough for practical purposes. Links to his solution are listed below:

- <https://irrational.net/2014/03/23/reverse-engineering-a-ceiling-fan/>
- <https://github.com/argilo/sdr-examples>

Smith's GNU Radio Flow Graph for Transmitting New Remote Signals

The article also lists the bit patterns he observed for the different buttons, which can be used for comparison later on:

- off: 0110100000010
- low: 0110100001000
- med: 0110100010000
- high: 0110100100000
- light: 0110100000001

There is another video/article using an RTL dongle, Spektrum (a great tool for fast scanning with an RTL), and Universal Radio Hacker. The remote in this article is very similar to mine but has extra address bits. Links to this example are listed below:

- <https://www.rtl-sdr.com/hacking-a-ceiling-fan-radio-control-signal-with-an-rtl-sdr/>
- <https://www.riveducha.com/decode-wireless-signal-with-usb-tv-tuner>

It seems that many others have already explored this topic, so there may be more code and articles online that I can reference if I need to uncover something more about how these messages work. However, I did not find a complete solution that I could reuse for demodulation or transmission. Neither of the remotes was detected by an older version of rtl_433 when tuned to their frequencies.

DEMODULATION

At this point, we know nearly everything about these remote controls, except for the meaning of the bits. After visually inspecting the recorded signals from both remotes, we can list them as follows, with the differing fields highlighted:

Old Remote (Jumpers 1-3 Set)

Fan High: 0011101110100

Fan Medium: 0011101110010

Fan Low: 0011101110001

Fan Off: 0011101111001

Light/Dimmer: 0011101111000

Reverse Direction: 0011101111010

New Remote (Five Switches: 1011-1)

Fan High: 0110111100000

Fan Medium: 01101111010000

Fan Low: 0110111001000

Fan Off: 0110111000010

Light/Dimmer: 0110111000001

Reverse Direction: 0110111000100

New Remote (Five Switches: 0100-1)

Light/Dimmer: 0101001000010

New Remote (Five Switches: 0100-0)

Light/Dimmer: 0101000000010

Studying these sequences reveals that the button data appears near the end of the message, while the address bits are towards the front. The new remote includes a bit for the “Dimmer/ON” switch after the address code. The purpose of the leading bits remains unclear, but they could be for synchronization, additional addressing, or distinguishing between models. Since there is no further functionality in the fan that need to explore, I am content leaving a bit of mystery.

I could easily build GNU Radio flow graphs to demodulate these signals and extract the bits, button type, address codes, and dimmer/on setting. However, I’ll leave that for a future lesson and plan to integrate it into FISSURE (possibly as an out-of-tree module on GitHub). The same applies to transmitting signals, but in the next section, I will cover an alternative method in FISSURE for transmitting OOK signals.

INJECTION

Technically, our research led us to someone who has already created a flow graph that successfully transmits the bits for my fan. Rather than recreating it, I’ll demonstrate the “IQ Data - OOK” tab, which serves as a generic signal generator for OOK signals. There is also a “General” FISSURE attack that includes a custom

FISSURE Dashboard

File View Options Standalone Tools Lessons Help

FISSURE THE RF FRAMEWORK Local Sensor Node New Sensor Node Stop

Automation TSI PD Attack **IQ Data** Archive Sensor Nodes Library Log

Record Playback Inspection Crop Convert Append Transfer Timeslot Overlap Resample OFDM

Normalize Strip Split **OOK**

On-Off Keying Signal Generation

Chip '0' Pattern: 010 Chip '0' Duration (μs): 3 Chip/Bit Sequence: 0110111000010

Chip '1' Pattern: 110 Chip '1' Duration (μs): 3

Burst Interval (μs): 50 Number of Bursts: 3

Sample Rate (MS/s): 1 Data Type: Complex Float 32

Plot Save

Open

Data Type: Complex Float 32

Choose: iling_fan_data/new

new_304_off.iq
new_304_off_cropped.iq
new_304_off_cropped_cropped.iq
new_304_off_cropped_cropped_cr
new_304_off_simulated.iq

Refresh Load File Delete

Rename Terminal SigMF

File Information

File:
Size:
Samples:

Sample Rate (MSps):
Frequency (MHz):

Plot

Plot All
Plot Range

Start:
End:

Cursor

Cursor Get Range

Filter

lowpass

Start (Hz): 100000
End (Hz): 200000

Filter

Functions

Mag. Phase

IF FFT

Spectrogram

Custom

Amplitude (LSB)

0.0 0.2 0.4 0.6 0.8 1.0

0 10000 20000 30000 40000 50000 60000 70000 80000

Samples

Moving Avg. Abs. Val. Differential Keep 1 in 2 Unwrap

Server: ipc://fissure Shutdown HIPRFSR: OK TSI: OK PD: OK SN1: OK SN2: - SN3: - SN4: - SN5: -

GNU Radio block, which translates the parameters from this tab into a flow graph, allowing it to be included in attack playlists.

Our new remote signals can be recreated in this tab from the following values acquired in our analysis:

- Chip "0" Pattern: 010
- Chip "1" Pattern: 110
- Burst Interval (μs): 13850
- Sample Rate (MS/s): 1
- Chip "0" Duration (μs): 343.333 (1030 μs/3)
- Chip "1" Duration (μs): 343.333
- Number of Bursts: 3 (minimum)
- Data Type: Complex Float 32
- Chip/Bit Sequence: 0110111000010 (13 bits, Fan Off)

OOK Signal Generator for Simulating the New Remote



An abstract graphic on the left side of the page, featuring a solid red vertical bar at the top left, a black silhouette of a person's head and shoulders below it, and a red, textured, curved shape resembling a stylized eye or a drop in the center.

Nihilistic

Magama
Bazarov

Nihilistic: Automated Cisco IOS Security Analysis

The issue of Cisco equipment security is especially acute, as the equipment of this vendor is widely spread all over the world. In this article we will talk about automated auditing of Cisco IOS devices using my utility “Nihilist”



Caster - Nihilistic

Genre: Defensive

Label: Medium

Release Date: 28 March 2025

Performed by: Caster

Written by: Magama Bazarov

Cover Man: Magama Bazarov (Sony ILCE-7M3, f/2.8, 1/200 sec)

Intro

This article is about automated security auditing of Cisco IOS network devices using the Nihilist tool. The purpose of the tool is to identify and analyze configuration vulnerabilities without interfering with device operation or applying attacking techniques. Nihilist runs over SSH using the netmiko library and does not modify the hardware configuration, using read-only commands with read-only privilege level commands.

Disclaimer

Nihilist is intended solely for analyzing Cisco IOS device configurations to identify potential vulnerabilities and security bugs. The tool is not intended and should not be used to:

- Conducting attacks or exploits;
- Disrupt network equipment;
- Modify or delete an existing configuration;
- Unauthorized access to devices;

The user is solely responsible for compliance with legal requirements, internal regulations and rules of use of the tool. The developer is not responsible for any consequences of improper or incorrect use of Nihilist.

Before running Nihilist, make sure that the account used to connect has read-only configuration permissions (`read-only`). This tool does not need an account that has permissions to change the configuration, write to NVRAM, etc.

Basic Concept

Nihilist is based on a simple architecture based on SSH connectivity and the use of regular expressions. Nihilist is written in Python and does not require additional software to be installed on Cisco devices.

What is Netmiko and why do I need it?

Nihilist uses the netmiko library, a popular Python framework for automating network devices. Netmiko allows you to establish SSH connections to Cisco equipment, as well as send commands to devices and receive responses in a convenient way. This library is chosen for its reliability, ease of use, and broad support for network devices, including Cisco IOS.

SSH Connection

An example of connecting Nihilist to a Cisco IOS device over SSH using Netmiko:

```
# Connect to the Cisco IOS
def connect_to_device(ip, username, password, port, device_type):
    print(Fore.WHITE + f"[*] Running on Python {sys.version.
split()[0]}" + Style.RESET_ALL)
    device = {
        "device_type": "cisco_ios",
        "host": ip,
        "username": username,
        "password": password,
        "port": port,
        "timeout": 10,
    }
    try:
        print(Fore.GREEN + f"[*] Connecting to {device_type} at
{ip}:{port}..." + Style.RESET_ALL)
        connection = ConnectHandler(**device)
        print(Fore.WHITE + "[*] Connection successful!\n" + Style.
RESET_ALL)
        return connection
    except NetmikoAuthenticationException:
        print(Fore.RED + "[-] Authentication failed! Check your
credentials." + Style.RESET_ALL)
        exit(1)
    except NetmikoTimeoutException:
        print(Fore.RED + "[-] Connection timed out! Check device
availability." + Style.RESET_ALL)
        exit(1)
    except Exception as e:
        print(Fore.RED + f"[-] Connection failed: {e}" + Style.RE-
SET_ALL)
        exit(1)
```

Using read-only commands (show)

A special feature of Nihilist is that the tool applies only `show` type commands, which completely eliminates the risk of changing the configuration of the tested equipment. This is important because the tool is designed as a diagnostic tool to identify weaknesses and possible threats without tampering with the device configuration.

Function Breakdown

Let's take a detailed look at how the DHCP Snooping analysis feature of the Nihilist tool works so that you can understand exactly how the tool works. I think this is very important when you are introducing your tool to everyone and a deep understanding is key.

What is DHCP Snooping?

DHCP Snooping is a Cisco IOS security mechanism that prevents attacks associated with rogue DHCP servers. DHCP Snooping inspects DHCP traffic and restricts it based on predefined criteria. Without DHCP Snooping enabled, the network is susceptible to:

- ### Stages of the function

The `checking_dhcp_snooping()` function implements the following steps:

1. **Getting the current configuration**
The command used is:

```
show running-config | section dhcp snooping
```

The result is output to the `dhcp_snooping_config` variable
3. If the result is blank, DHCP Snooping is not configured!
4. **Checking if DHCP Snooping is enabled**
5.

```
if not dhcp_snooping_config:  
    print("[!] WARNING: DHCP Snooping is NOT enabled on this de-  
vice!")
```

This is a critical warning if the mechanism is completely off.
6. **Analyzing the VLAN configuration**
Using a regular expression, checks which VLANs have DHCP Snooping enabled:
7.

```
snooping_vlans = re.findall(r'ip dhcp snooping vlan (\S+)',  
dhcp_snooping_config)
```
8. If VLANs are not explicitly specified, a warning is generated
9. **Checking trusted ports**
Ports on which trusted DHCP traffic passes without restriction:
10.

```
snooping_trust_ports = re.findall(r'ip dhcp snooping trust',  
dhcp_snooping_config)
```
11. If trusted ports are not available, DHCP traffic will be blocked, which will disrupt the network
12. **Option 82 (DHCP Snooping Information Option)**
Enabled by default, may cause compatibility issues with DHCP servers:
13.

```
option_82_disabled = "no ip dhcp snooping information option"  
in dhcp_snooping_config
```

14. It is recommended to disable Option 82 on most networks. However, it is worth considering on a case-by-case basis, as Option 82 is sometimes used by network engineers

15. **DHCP Snooping Database Configuration**

Checks if the database is configured to store DHCP bindings:

```
16. snooping_database = re.search(r'ip dhcp snooping database\s+(-flash:|ftp:|https:|rcp:|scp:|tftp:)/\S+', dhcp_snooping_config)
```

17. If not configured, all DHCP bindings will be lost after a reboot

18. **Write-delay for DHCP Snooping database**

Interval for writing changes to the database:

```
19. write_delay_match = re.search(r'ip dhcp snooping database write-delay (\d+)', dhcp_snooping_config)
```

20. If this parameter is missing, changes may not be recorded efficiently and data may be lost

21. **Rate Limit**

Checks if there is a rate limit on DHCP requests:

```
22. rate_limit = re.findall(r'ip dhcp snooping limit rate (\d+)', dhcp_snooping_config)
```

23. This is important to prevent DHCP Exhaustion

This is exactly how “Nihilist” performs Cisco IOS security analysis: obtaining configuration data and parsing it using regular expressions.

What does Nihilist check?

“Nihilist” performs a detailed configuration audit of Cisco devices, analyzing various security aspects. The entire audit is divided into multiple stages.

System & Operational Checks

These checks help you quickly understand the status of the device and its configuration at the most basic level:

- **Uptime Check:** Checks how long the device has been running without rebooting. A value that is too high (years) may indicate that the device has not been updated and is vulnerable.
- **Configuration Size Estimation:** Analyzes the size of the current configuration. A configuration that is too large requires closer attention and may contain unwanted settings or outdated policies.

IOS Security Analyzing

This category includes checks for general device security settings, including access control, authentication, and cryptography.

- **PAD Service Check:** Checks if the Packet Assembler/Disassembler service is enabled, which can be used by an attacker to compromise the device;
- **Service Password Encryption Check:** Checks if basic password encryption is present and used in the configuration;
- **Password Hashing Check:** Determines the type of password hashing used (types 4/5/7/8/9) and identifies weak algorithms;
- **RBAC Check:** Analyzes role-based access control settings and identifies potential minimum privilege violations;
- **VTY Security Check:** Checks the security of virtual terminals (Telnet, SSH), access-class restric-

tions and potential vulnerabilities;

- **AAA Check:** Analyzes the presence and correctness of centralized access control settings;
- **Session Limit Check:** Evaluates whether there is a limit on the number of concurrent user sessions;
- **Login Block Protection Check:** Checks for the presence of a login block-for protection mechanism to prevent brute force attacks;
- **SSH Security Check:** SSHv2 usage, timeout setting, and other key parameters;
- **Default Usernames Check:** Search for default or weak accounts (cisco, admin, root) that can be easily compromised;
- **SNMP Security Check:** Check SNMP community strings for weak values (public/private);
- **Smart Install Check:** Check for Smart Install service activity, which is often used by attackers to take over a device.

L2 Security Analyzing

Here, Nihilist identifies common Layer 2 security issues, including VLAN manipulation and attacks on link layer protocols.

- **VTP Status Check:** Check the status of the VTP protocol, which can be used by attackers to modify the VLAN base;
- **DTP Status Check:** Checks if DTP is enabled on interfaces, which allows for a VLAN Hopping attack;
- **Native VLAN Check:** Analyzes Native VLAN usage to prevent double tagging attacks;
- **DHCP Snooping Check:** Check the state of protection against DHCP attacks (spoofing of DHCP servers);
- **DAI Check:** Check for protection against ARP Spoofing;
- **BPDU Guard Check:** Check protection against STP attacks (partial MITM, root roles hijacking);
- **Storm Control Check:** Analyzing protection against broadcast storms;
- **Port Security Check:** Check of MAC-address restrictions on ports (restriction of ports for unauthorized connection of devices)

L3 Security Analyzing

In this block, Nihilist inspects the security mechanisms of routing and redundancy protocols, identifying configuration errors and weak settings that can lead to network-level attacks.

- **HSRP Check:** Analyze HSRP configuration to identify risks of MITM attacks;
- **VRRP Check:** Analyze VRRP configuration to identify risks of MITM attacks;
- **GLBP Check:** Analyze GLBP configuration to identify risks of MITM attacks;
- **OSPF Passive Interfaces Check:** Checks OSPF interfaces for passive mode to prevent unauthorized routing;
- **OSPF Authentication Check:** Analyze OSPF authentication types (Null, MD5, SHA) to identify insufficient security;
- **EIGRP Passive Interfaces Check:** Checks EIGRP interfaces for passive mode to prevent unauthorized routing;
- **EIGRP Authentication Check:** Analyze the presence and reliability of authentication in EIGRP (Null, MD5, key-chains)

How to Use

To install the Nihilist:

```
:~$ sudo apt install git python3-colorama python3-netmiko
:~$ git clone https://github.com/casterbyte/Nihilist
:~$ cd Nihilist
:~/Nihilist$ sudo python3 setup.py install
:~$ nihilist --help
```

```
usage: nihilist.py [-h] --ip IP --username USERNAME --password
PASSWORD [--port PORT] [--router] [--l2-switch] [--l3-switch]
```

options:

-h, --help	show this help message and exit
--ip IP	Specify the IP address of the device
--username USERNAME	SSH Username
--password PASSWORD	SSH Password
--port PORT	SSH Port (default:22)
--router	Specify if the device is a router
--l2-switch	Specify if the device is a L2 switch
--l3-switch	Specify if the device is a L3 switch

Trigger Arguments (CLI Options)

Nihilist supports as input parameters:

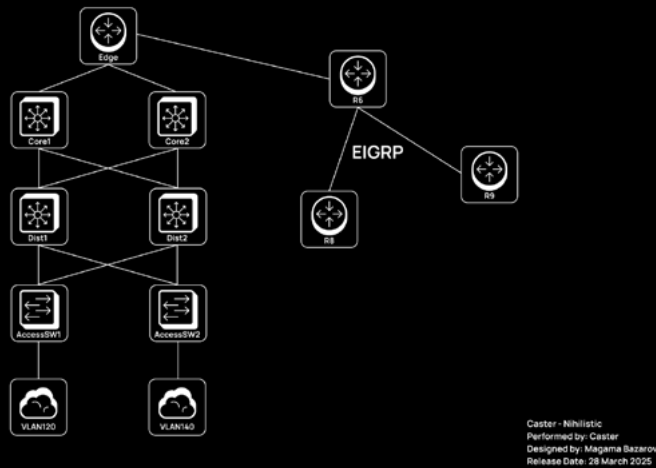
- `--ip`: the user will need to specify the IP address of their device;
- `--username`: the username for SSH connection to the Cisco device;
- `--password`: the password for SSH connection to the Cisco device;
- `--port`: SSH port number, by default the tool uses port 22;
- `--router`: if the Cisco device is a router;
- `--l2-switch`: if it's a Cisco L2 switch;
- `--l3-switch`: if it's a Cisco L3 switch.

Important

- Be sure to specify only ONE of the parameters: `--router`, `--l2-switch`, or `--l3-switch`, depending on the device type;
- If you do not specify a device type, the audit will not be performed;
- Make sure the account has access to at least the show commands, as "Nihilist" is read-only and does not require privilege level 15

Practical Security Analysis of a Three-Layer Network

In the previous steps, I went over the functionality of Nihilist and its ability to identify vulnerabilities in the network infrastructure. Now, to demonstrate how it works, we will perform a detailed security audit of a typical three-tier hierarchical network consisting of Core, Distribution, and Access layers.



The scheme above shows the network topology that we will use as a test bed for our analysis. Our goal is to apply Nihilist to test devices at each layer, identify potential vulnerabilities, and offer recommendations for remediation.

Edge

```
caster@kali:~$ nihilist --ip 172.16.0.2 --username caster --password caster --router
```

```

  _____
 /         \
|   Nihilist   |
 \         /
  _____

Nihilist: Cisco IOS Security Inspector
Author: Magama Bazarov, <magamabazarov@mailbox.org>
Alias: Caster
Version: 1.0
Codename: Gestalt
How to Use: https://github.com/casterbyte/Nihilist
Detailed Documentation: https://github.com/casterbyte/Nihilist/wiki/Mechanism-of-the-tool

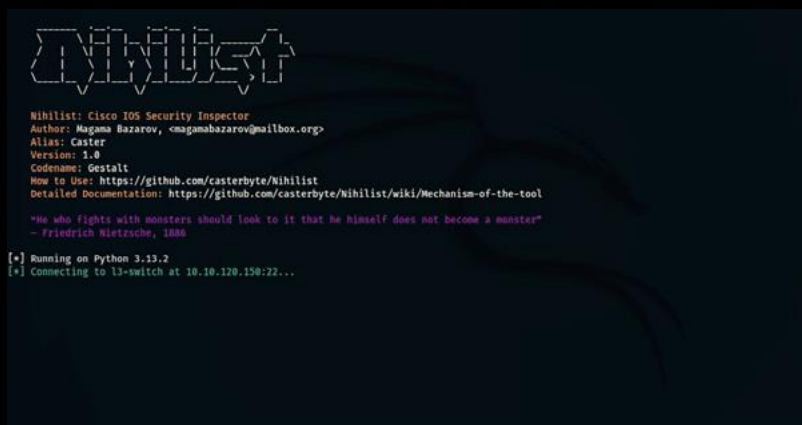
"Be who you fight with monsters should look to it that he himself does not become a monster"
- Friedrich Nietzsche, 1886

[*] Running on Python 3.13.2
[*] Connecting to router at 172.16.0.2:22...
```



WL3 Switch

```
caster@kali:~$ nihilist --ip 10.10.120.150 --username caster --password caster --l3-switch
```

A terminal window with a dark background. At the top, there is a logo for 'nihilist' made of circuit-like symbols. Below it, the tool's metadata is displayed: 'Nihilist: Cisco IOS Security Inspector', 'Author: Magama Bazarov, <magamabazarov@mailbox.org>', 'Alias: Caster', 'Version: 1.0', 'Codename: Gestalt', 'How to Use: https://github.com/casterbyte/nihilist', and 'Detailed Documentation: https://github.com/casterbyte/nihilist/wiki/Mechanism-of-the-tool'. A quote by Friedrich Nietzsche is shown: '"He who fights with monsters should look to it that he himself does not become a monster" - Friedrich Nietzsche, 1886'. At the bottom, two status messages are shown: '[*] Running on Python 3.13.2' and '[*] Connecting to l3-switch at 10.10.120.150:22...'.

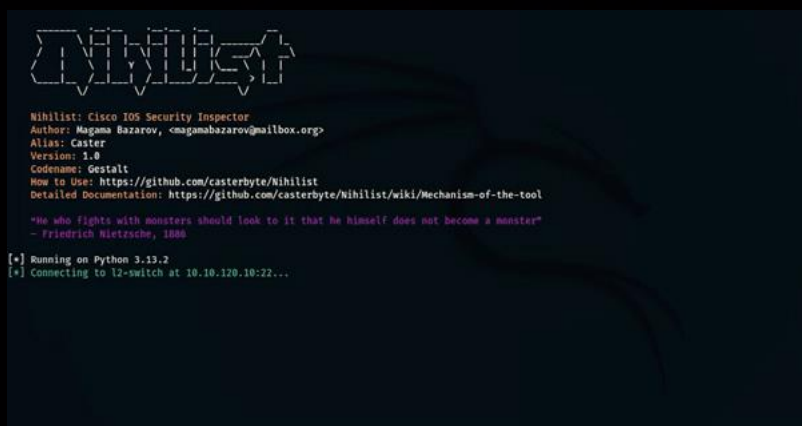
```
Nihilist: Cisco IOS Security Inspector
Author: Magama Bazarov, <magamabazarov@mailbox.org>
Alias: Caster
Version: 1.0
Codename: Gestalt
How to Use: https://github.com/casterbyte/nihilist
Detailed Documentation: https://github.com/casterbyte/nihilist/wiki/Mechanism-of-the-tool

"He who fights with monsters should look to it that he himself does not become a monster"
- Friedrich Nietzsche, 1886

[*] Running on Python 3.13.2
[*] Connecting to l3-switch at 10.10.120.150:22...
```

L2 Switch

```
caster@kali:~$ nihilist --ip 10.10.120.10 --username caster --password caster --l2-switch
```

A terminal window with a dark background. At the top, there is a logo for 'nihilist' made of circuit-like symbols. Below it, the tool's metadata is displayed: 'Nihilist: Cisco IOS Security Inspector', 'Author: Magama Bazarov, <magamabazarov@mailbox.org>', 'Alias: Caster', 'Version: 1.0', 'Codename: Gestalt', 'How to Use: https://github.com/casterbyte/nihilist', and 'Detailed Documentation: https://github.com/casterbyte/nihilist/wiki/Mechanism-of-the-tool'. A quote by Friedrich Nietzsche is shown: '"He who fights with monsters should look to it that he himself does not become a monster" - Friedrich Nietzsche, 1886'. At the bottom, two status messages are shown: '[*] Running on Python 3.13.2' and '[*] Connecting to l2-switch at 10.10.120.10:22...'.

```
Nihilist: Cisco IOS Security Inspector
Author: Magama Bazarov, <magamabazarov@mailbox.org>
Alias: Caster
Version: 1.0
Codename: Gestalt
How to Use: https://github.com/casterbyte/nihilist
Detailed Documentation: https://github.com/casterbyte/nihilist/wiki/Mechanism-of-the-tool

"He who fights with monsters should look to it that he himself does not become a monster"
- Friedrich Nietzsche, 1886

[*] Running on Python 3.13.2
[*] Connecting to l2-switch at 10.10.120.10:22...
```

Outro

Cisco equipment is very common all over the world and security is a major issue. Use it wisely and take care of the security of your infrastructure.



mr licka

open source frequencies

Cyber Security Labs

Why they are important?

Wireless Wizardry

DragonOS & other RF Shinanigans



three.twozerotwofour

PLINY

THE PROMPTER

CAPTAIN CRUNCH

AKA JOHN DRAPER

CHEMDELIC

CLANDESTINE

CHEMIST

LOUTSMARTER

DR. RICHARD

FEYNMAN

THE DUKE OF

DISTORTION

BOLLARD

TRYPTAMINES

A CURE FOR

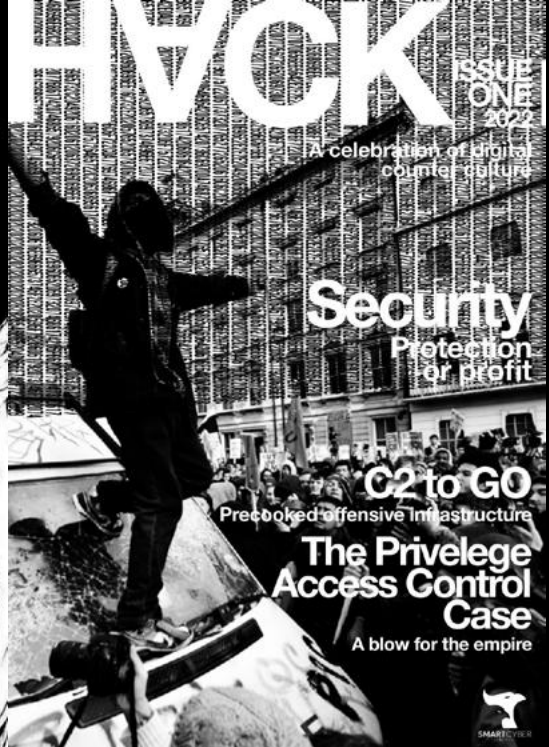
OPICIA ADDICTION

IN TECH?

HACKER

HERMANOS

C2 & TACTICAL PRETEXTING



Security
Protection
or profit

C2 to GO

Precooked offensive Infrastructure

The Privilege
Access Control
Case

A blow for the empire



hacker
hermanos

Red team infrastructure as code

1337
sheets

The significance of data sheets
in hardware hacking

ghost rider
in the machine

Build an AI powered pentesting copilot

anonymous
resources

Fly under the radar with these
amazing tools and services

issue:one:twozerotwofour

Smart Cyber Solutions Project



firmware
emulation

Reversing the firmware of a USB stick

the threat
inside

Understanding the insider threat with network forensics

osint in
the dark

Using Metasploit as an event tool on the darknet

exclusive
5g core exploit

hijacking network traffic with profile manipulation

A celebration of digital counter-culture

issue:twozerotwofour



five:twozerotwotwo

HACK: a celebration of digital counter-culture

osint
honours

the journey of a trace labs judge

feminist
hackerspaces

artificial intelligence
& future crimes

the potential fate of skynet

afterdark

art & images of Indigo
Perry, Marina Abramovic
& Fiona Lewis

the
wonderful women
cyber security issue



one:twozerotwotwo

HACK: a celebration of digital counter-culture

ice
man

the people's
king of rfid

dr oren
eytan

sensitive
data's saviour

pwning
epson

hacking
printers for
fun and profit

