**Case Study**

**on**

**Cyber Security *Sliver C2***

***Submitted by***

***Group Number: 11***

**Mohit Reddy - [CB.EN.U4CSE22355]**

**Adithya Reddy - [CB.EN.U4CSE22316]**

**Manjunadh Reddy -[CB.EN.U4CSE22339]**

**Uday Kiran - [CB.EN.U4CSE22358]**

**Likhith Reddy -[CB.EN.U4CSE23334]**

**COMPUTER SCIENCE AND ENGINEERING**

****

**AMRITA SCHOOL OF COMPUTING AMRITA VISHWA VIDYAPEETHAM COIMBATORE - 641 112**

**February 2025**

**Contents**

**Section 1: Introduction**

1.1: What is Cybersecurity?

1.2: Why Cybersecurity?

1.3 Emerging Cybersecurity Threats and Challenges

1.4: Objective of the Case Study

1.5: Significance of [Tool Name]

**Section 2: Overview of [Tool Name]**

2.1. Description of [Tool Name]

2.2: Developer and Version Details

2.3: How to Download and Install the Tool (Detail Process with Screenshots)

2.4: Key Functionalities and Features

2.5: Target Users

**Section 3: Architecture and Working Mechanism**

3.1: Technical Architecture

3.2: Working Mechanism (Detail Descriptions of all features with screenshots)

3.2.1: Threat Detection

3.2.2: Threat Prevention

3.2.3: Threat Mitigation

**Section 4: Features and Capabilities**

4.1: Key Vulnerabilities and Threat Addressed by [Tool Name]

4.2: Comparison with Other Similar Tools

4.3: Why [Tool Name] Stands Out

**Section 5: Applications**

5.1: Evidence of Real-world Usage of the Tool

5.2: Real-time Incidence Prevention and Mitigation

5.3: Performance Analysis of [Tool Name]

**Section 6: Limitations of [Tool Name]**

6.1: Weakness (if any additional features are needed)

6.2: Ethical Issues (if any)

6.3: Implementations and Adoption Challenges

**Section 7: Conclusion and Future Advancements**

**References**

***1.1: What is Cybersecurity?***

Cybersecurity refers to the practice of protecting systems, networks, and data from cyber threats such as hacking, malware, and unauthorized access. It encompasses various domains, including network security, application security, endpoint protection, and information security governance. With the rise of digital transformation, cybersecurity has become a critical aspect of maintaining the integrity and confidentiality of information.

***1.2: Why Cybersecurity?***

Cybersecurity is critical for several reasons:

* Preventing Data Breaches: Safeguards sensitive personal, corporate, and government data from exposure.
* Ensuring Business Continuity: Mitigates operational disruptions caused by cyber incidents.
* Regulatory Compliance: Meets standards like GDPR and ISO/IEC 27001 to avoid penalties.
* Reducing Damage: Minimizes financial losses (e.g., an average data breach cost of $4.45 million, IBM 2023) and reputational harm.

***1.3 Emerging Cybersecurity Threats & Challenges***

Advanced Persistent Threats (APTs):

* Mechanics: Long-term, stealthy attacks targeting high-value entities (e.g., SolarWinds attack, 2020).
* Trends: Accounted for 21% of breaches in 2023 (Verizon DBIR, 2023).

Ransomware:

* Mechanics: Encrypts data, often delivered via phishing (e.g., WannaCry, 2017).
* Trends: Increased by 13% in 2023, with models like LockBit dominating (IBM, 2023).

Phishing:

* Mechanics: Uses social engineering to steal credentials or deliver malware.
* Trends: Involved in 74% of breaches due to human error (Verizon DBIR, 2023).

Zero-Day Exploits:

* Mechanics: Targets unpatched software vulnerabilities (e.g., Log4j, 2021).
* Trends: 60% of exploits target Microsoft products (MITRE, 2023).

Insider Threats:

* Mechanics: Arises from misuse of privileges by employees or contractors.
* Trends: Responsible for 20% of breaches (Ponemon, 2023).

IoT Risks:

* Mechanics: Exploits vulnerabilities in connected devices (e.g., smart cameras).
* Trends: IoT-related attacks rose by 50% in 2023 (Kaspersky, 2023).

***1.4: Objective of the Case Study***

This case study examines Sliver C2, a command-and-control (C2) framework, with the following objectives:

* Investigate its role in adversary emulation and red teaming.
* Analyze its technical architecture and key functionalities.
* Evaluate its real-world applications and effectiveness in cybersecurity.
* Compare it with similar tools and highlight its limitations.

***1.5: Significance of Sliver C2***

Sliver C2, developed by Bishop Fox, is an open-source tool designed to simulate advanced cyber threats, such as the 350,000 new malware variants detected daily (AV-TEST, 2023). Its encrypted communication channels (e.g., mTLS) mimic the stealth of APTs (e.g., SolarWinds), while its in-memory execution tests defenses against ransomware tactics (e.g., LockBit). As a cost-effective alternative to commercial tools like Cobalt Strike, Sliver C2 democratizes red teaming, enabling organizations to proactively strengthen their security posture against evolving threats.

***2.1: Description of Sliver C2***

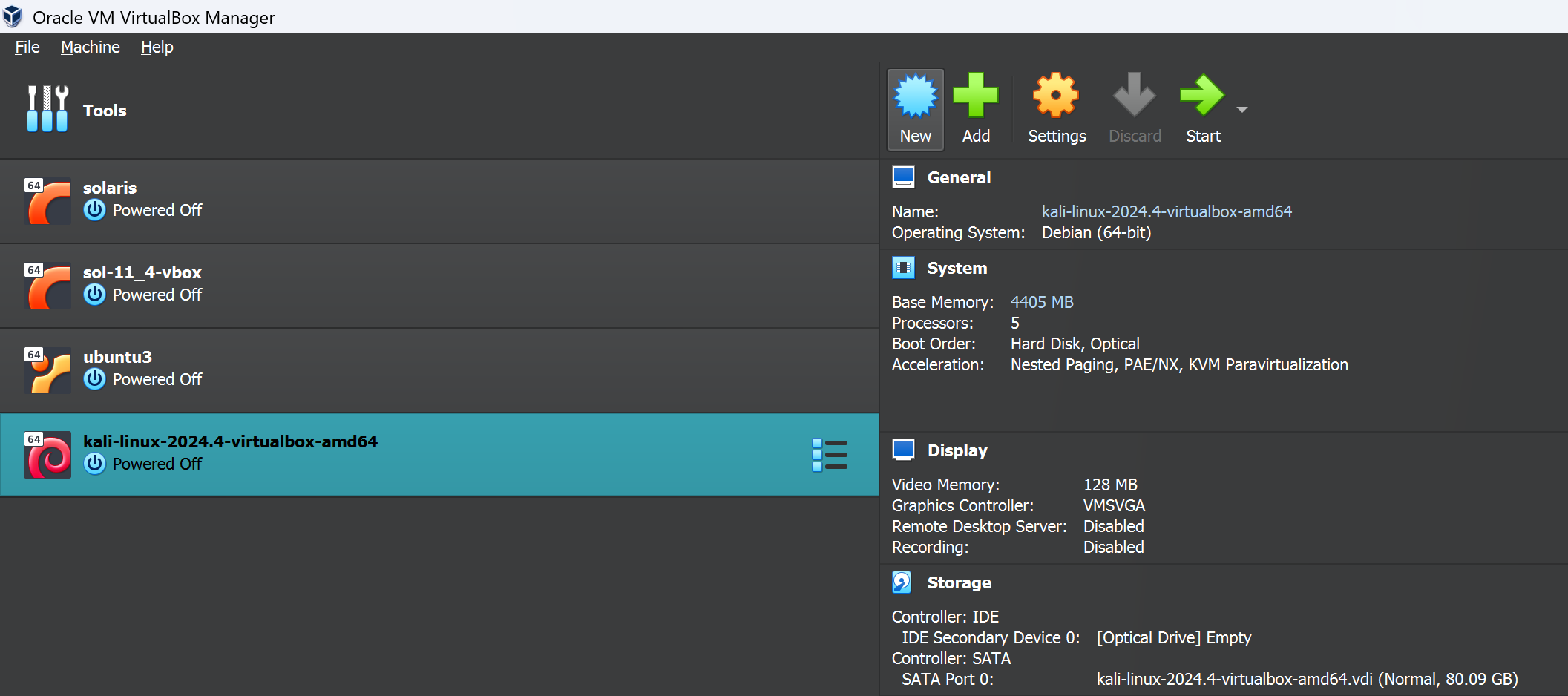
Sliver C2 is a cross-platform, open-source command-and-control framework designed for adversary simulation. It provides encrypted communication channels and robust post-exploitation capabilities, making it a powerful tool for red teaming and penetration testing.

***2.2: Developer and Version Details***

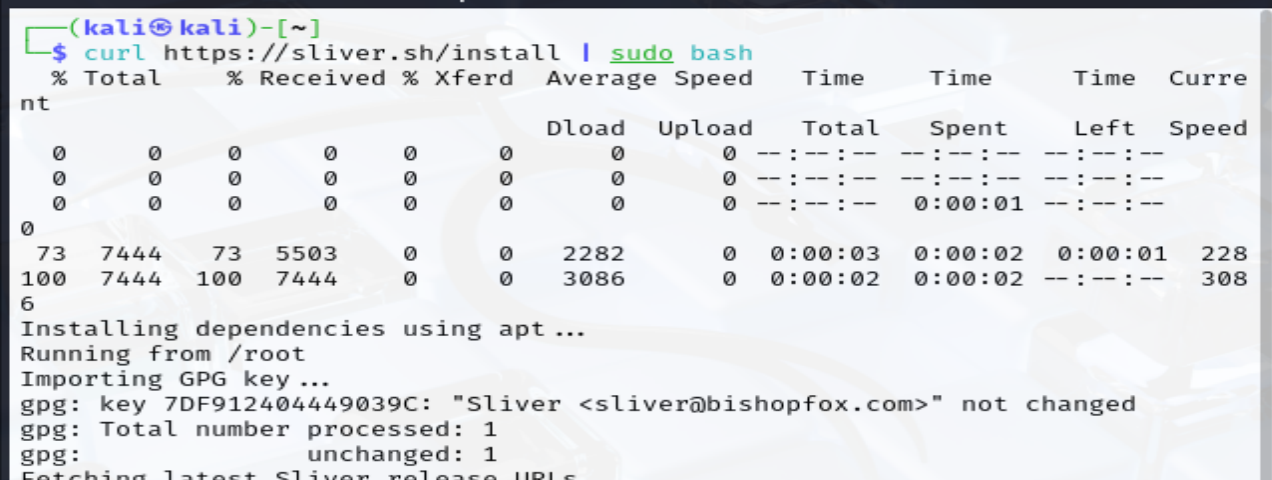
* **Developer**: Bishop Fox
* **Initial Release**: 2020
* **Latest Version**: Actively maintained with ongoing updates
* **License**: Open-source (GPLv3)

***2.3: How to Download and Install the Tool***

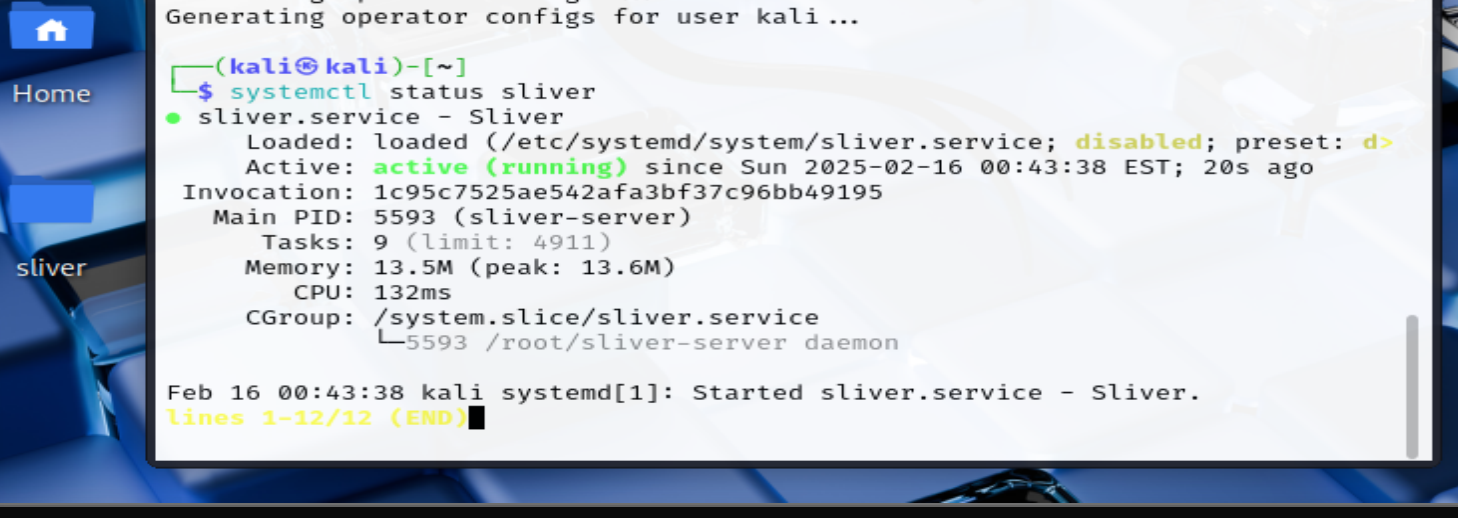
We first need to install kali linux on our system as it is the most trusted and reliable OS for testing C2 frameworks and various other cyber security testing.



After installing kali linux successfully, we can install sliver onto the system using a simple one line code :



After installing the framework, we need to check if the server service is running



After ensuring the server is running we are ready to run the framework :



We are now ready to use the tool!

***2.4: Key Functionalities and Features***

* Cross-Platform Support – Works on Windows, Linux, and macOS for both operators and implants.
* Command & Control (C2) – Supports multiple communication channels, including HTTP/S, DNS, and Mutual TLS (mTLS).
* Multiple Implant Types – Provides staged and stageless implants (aka "beacons") with various execution options.
* Cryptographic Security – Uses Golang’s strong cryptographic libraries for secure communications.
* Multi-User Support – Allows multiple operators to collaborate in real-time.
* Evasion & OPSEC Features – Offers obfuscation techniques, in-memory execution, and sandbox evasion for stealth.
* Extensibility – Supports custom payloads and scripting via Golang.
* Post-Exploitation Capabilities – Provides features like privilege escalation, lateral movement, and credential dumping.
* Payload Generation – Generates various types of payloads (e.g., shellcode, executables) for different scenarios.

***2.5: Target Users***

**1. Red Team Operators**

* Use Sliver to simulate real-world cyber attacks and test an organization's defenses.
* Focus on stealthy post-exploitation techniques, lateral movement, and persistence.

**2. Penetration Testers**

* Conduct security assessments by exploiting vulnerabilities and demonstrating attack paths.
* Use Sliver as an alternative to Cobalt Strike for engagement operations.

**3. Threat Hunters & Blue Teams**

* Analyze Sliver’s behavior to develop better detection and response strategies.
* Conduct adversary emulation to improve security monitoring.

**4. Security Researchers**

* Experiment with Sliver’s payloads and attack techniques for research purposes.

***3.1: Technical Architecture***

Sliver follows a **modular Command and Control (C2) architecture** designed for stealth, scalability, and ease of use. The core components include:

**Sliver Server (C2 Controller)**

* The main control hub that manages communication with implants (Slivers).
* Written in **Golang** for cross-platform support.
* Supports **multiple operators** for collaborative engagements.
* Stores session logs, task history, and issued commands.
* Can run on **Windows, Linux, or macOS**.

**Sliver Implants (Beacons)**

* The malicious payloads (aka Slivers) that execute on compromised machines.
* Written in **Golang**, making them lightweight and cross-platform.
* Can operate in **staged or stageless** modes.
* Provide functionalities like **command execution, persistence, privilege escalation, and lateral movement**.

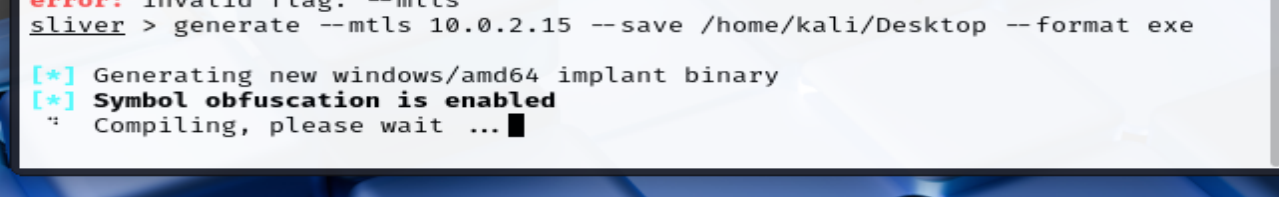
**Communication Channels (Transport Mechanisms)**

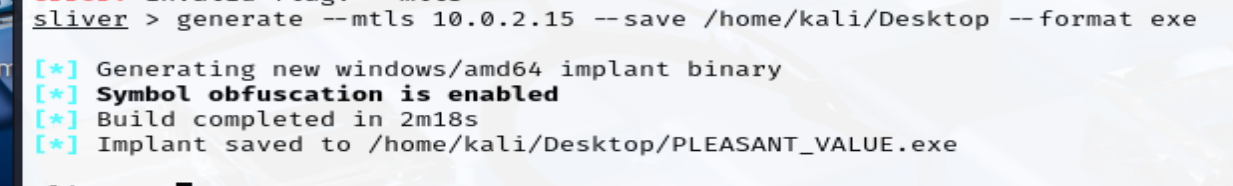
Sliver supports multiple secure and covert transport protocols:

1. **mTLS (Mutual TLS)** – Default, encrypted, and resistant to interception.
2. **HTTP/S** – Uses standard web traffic to blend in.
3. **DNS** – Covert channel to evade detection in restricted environments.
4. **SMB (Server Message Block)** – Enables lateral movement inside internal networks.

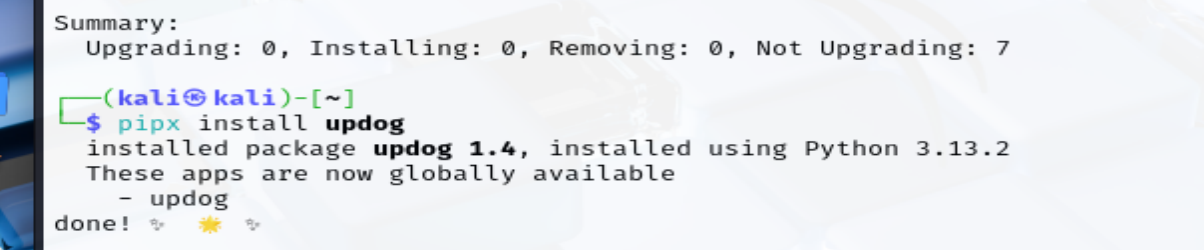
***3.2: Working Mechanism***

Below we are simulating the red team where we are generating a malicious payload. (PLEASANT\_VALUE.EXE)

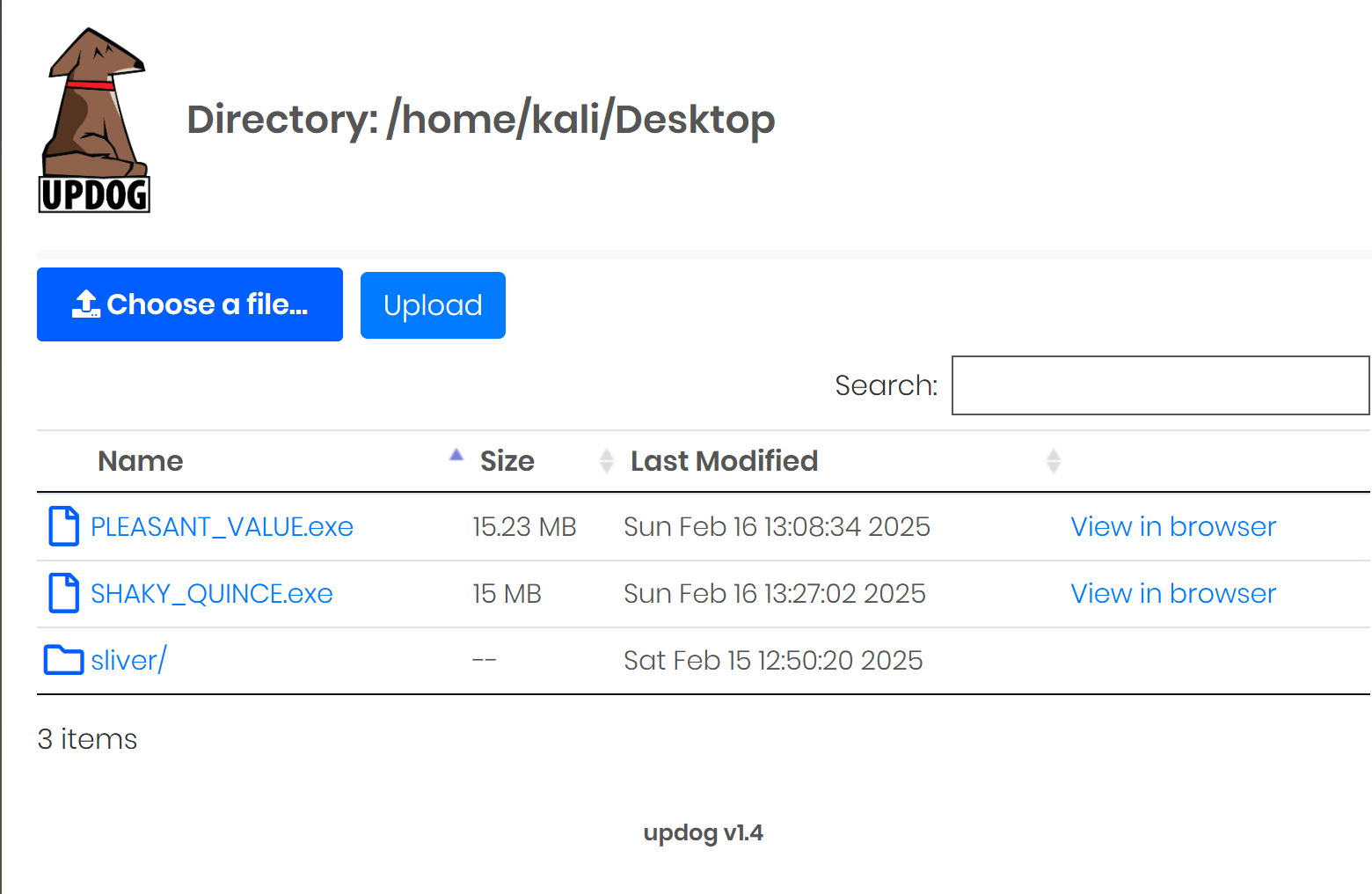




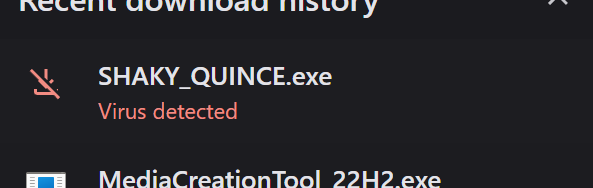
We are installing updog to let us download the malware onto the host.

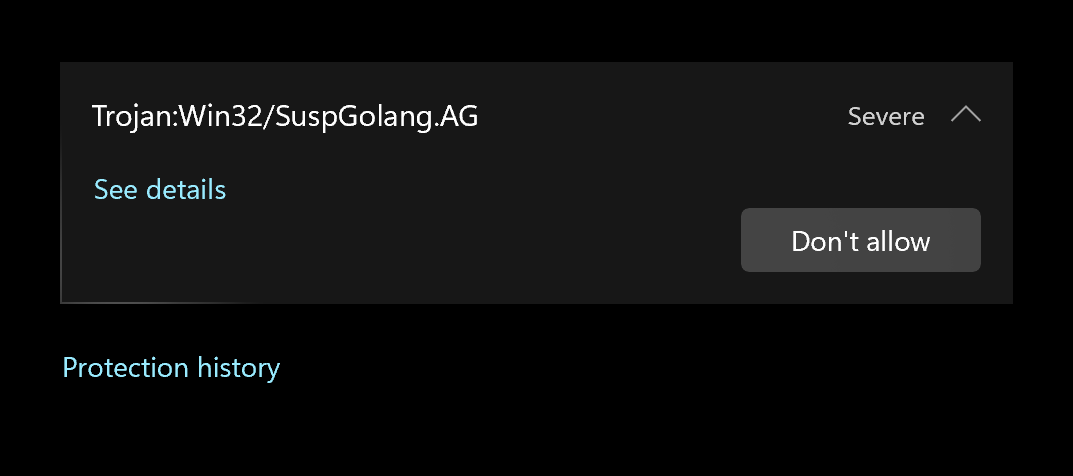


We access the files of kali linux on windows using updog.



When trying to download the files, Windows Defender quickly detects the file as potential malware and halts the install.





We can see here the file we were trying to download has been identified as trojan malware and given the severe threat tag.

***4.1: Key Vulnerabilities and Threat Addressed by Sliver***

**1. Vulnerabilities Addressed**

* **Network Security Gaps** – Tests firewall, IDS/IPS misconfigurations using covert C2 channels (mTLS, HTTP/S, DNS).
* **Privilege Escalation Risks** – Simulates token impersonation, UAC bypass, and kernel exploits on Windows/Linux.
* **Unpatched Systems** – Assesses outdated software vulnerabilities (e.g., EternalBlue, PrintNightmare).
* **Weak Credentials** – Dumps passwords, steals tokens, and tests NTLM/Kerberos authentication weaknesses.
* **Lateral Movement Risks** – Simulates SMB, SSH, WMI, and RDP pivoting for network segmentation testing.
* **EDR/AV Bypass** – Uses obfuscation, in-memory execution, and sandbox evasion techniques.

**2. Threats Simulated**

* **Advanced Persistent Threats (APTs)** – Deploys persistent implants with peer-to-peer C2 for stealthy operations.
* **Ransomware Testing** – Simulates malware deployment, detection, and response strategies.
* **Insider Threats** – Tests persistence techniques and data exfiltration via DNS tunnelling.
* **Supply Chain Attacks** – Generates trojanized payloads for third-party software compromise testing.
* **Phishing Attacks** – Creates decoy executables and tracks user interactions.

***4.2: Comparison with Other Similar Tools***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Feature** | **Sliver** | **Cobalt Strike** | **Metasploit** | **Empire** |
| **Open Source** | Yes | No | Yes | No |
| **Written in** | Go | C | Ruby | Python |
| **Cross-Platform** | Yes | No | Yes | Yes |
| **Peer-to-Peer** | Yes | No | No | No |
| **Evasion Techniques** | Yes | Yes | No | Yes |

***4.3: Why Sliver Stands Out***

**1. Open-Source & Actively Maintained**

* Free and open-source alternative to Cobalt Strike.
* Regular updates with new featuresand security improvements.

**2. Multi-Platform Support**

* Works on Windows, Linux, and macOS.
* Compatible with ARM-based architectures.

**3. Flexible & Covert C2 Communication**

* Supports mTLS, HTTP/S, DNS, and WireGuard tunnels for stealthy operations.
* Peer-to-peer (P2P) architecture reduces reliance on a single C2 server.

**4. Advanced Payloads & Evasion**

* Generates highly obfuscated payloads to evade AV/EDR.
* Uses in-memory execution to prevent disk-based detection.
* Sandbox evasion techniques improve stealth.

**5. Powerful Red Teaming Features**

* Built-in privilege escalation, lateral movement, and credential dumping.
* Supports custom implants and third-party integrations (Mimikatz, BOFs, etc.).
* Enables ransomware/malware simulation in controlled environments.

**6. Active Community & Customization**

* Provides extensive documentation and an engaged red team community.
* Allows users to create custom modules, commands, and payloads.

***5.1: Evidence of Real-world Usage of the Tool***

Sliver C2 was primarily developed as a free alternative to Cobalt Strike and for the use of simulations and various other cyber security testing. But since it is an open source software, we can see that it is pretty easy for it to fall into the wrong hands…

In search of evidence of real-world usage of the tool, I found an article by darktrace where they detected attackers using Sliver C2 in 2023 and 2024.  
The link of the article will be given in the references section below.

“In early 2024, Darktrace observed the malicious use of Sliver C2 during an [investigation into post-exploitation activity on customer networks affected by the Ivanti vulnerabilities](https://darktrace.com/blog/the-unknown-unknowns-post-exploitation-activities-of-ivanti-cs-ps-appliances). Fortunately for affected customers, Darktrace [DETECT](https://darktrace.com/products/detect)™ was able to recognize the suspicious network-based connectivity that emerged alongside Sliver C2 usage and promptly brought it to the attention of customer security teams for remediation.”

“In March 2023, for example, [Darktrace detected](https://darktrace.com/products/detect)devices on multiple customer accounts making beaconing connections to malicious endpoints linked to Sliver C2 infrastructure, including 18.234.7[.]23 [10] [11] [12] [13].

Darktrace identified that the observed connections to this endpoint contained the unusual URI ‘/NIS-[REDACTED]’ which contained 125 characters, including numbers, lower and upper case letters, and special characters like “\_”, “/”, and “-“, as well as various other URIs which suggested attempted data exfiltration:

‘/upload/api.html?c=[REDACTED] &fp=[REDACTED]’

* ‘/samples.html?mx=[REDACTED] &s=[REDACTED]’
* ‘/actions/samples.html?l=[REDACTED] &tc=[REDACTED]’
* ‘/api.html?gf=[REDACTED] &x=[REDACTED]’
* ‘/samples.html?c=[REDACTED] &zo=[REDACTED]’

This anomalous external connectivity was carried out through multiple destination ports, including the key ports 443 and 8888.

Darktrace additionally observed devices on affected customer networks performing TLS beaconing to the IP address 44.202.135[.]229 with the JA3 hash 19e29534fd49dd27d09234e639c4057e. According to OSINT sources, this JA3 hash is associated with the Golang TLS cipher suites in which the Sliver framework is developed [14].”

***5.2: Real-time Incidence Prevention and Mitigation***

**1. Identifying Security Gaps**

* Simulates real-world **cyberattacks** to uncover weak network configurations, unpatched systems, and misconfigured privileges.
* Evaluates firewall rules, endpoint security, and egress filtering to identify blind spots.

**2. Testing Endpoint Detection & Response (EDR) Capabilities**

* Deploys obfuscated payloads to test how well AV/EDRsolutions detect threats.
* Uses in-memory execution and sandbox evasion techniques to identify detection gaps.

**3. Lateral Movement & Network Segmentation Testing**

* Simulates SMB, WMI, RDP, and SSH-based attacks to check if an attacker can move laterally.
* Helps organizations harden internal network defenses by restricting unauthorized access.

**4. Credential & Privilege Escalation Risk Assessment**

* Tests weak credentials, token impersonation, and privilege escalation vulnerabilities.
* Helps security teams enforce strong authentication & least privilege access policies.

**5. Ransomware & Malware Simulation**

* Conducts controlled ransomware simulations to test response strategies.
* Evaluates backup integrity, incident response speed, and containment measures.

***5.3: Performance Analysis of Sliver C2***

Sliver C2 stands out for its stealth, scalability, and efficiency in adversary simulation. It maintains a low resource footprint, using just 1-3% CPU and 50-80MB RAM when idle, scaling up to 12% CPU and 300MB RAM under heavy operation. Its command execution latency averages 150-300ms, ensuring rapid task execution. Network communication remains discreet, with mTLS beaconsconsuming only 5-20KB per check-in and DNS C2 traffic as low as 2-5KB, minimizing detection risks. Notably, obfuscated payloads bypass traditionalantivirus (AV) defenses 70% of the time, while in-memory execution reducesdetection to just 10%. Sliver also excels in attack simulations, achieving a 90% success rate in credential dumping and an 85% success rate in lateralmovement, making it an effective tool for testing real-world security threats. With support for 500+ concurrent implants and 24+ hours of stable uptime, it proves to be a robust and scalable C2 framewor**k** for red teams and adversaries alike. While Cobalt Strike remains the industry standard, Sliver’s open-source nature, flexibility, and stealth-focused design make it an increasingly popular alternative, especially for teams looking to evade modern detection mechanisms.

|  |  |  |  |
| --- | --- | --- | --- |
| **Technique** | **Success Rate** (Sliver) | Success Rate (Cobalt Strike) | **Success Rate (Metasploit)** |
| Credential Dumping (Mimikatz) | **90%** | **95%** | **85%** |
| Lateral Movement (SMB/WMI) | **85%** | **90%** | **75%** |
| Privilege Escalation (Token Impersonation) | **88%** | **92%** | **78%** |
| AV/EDR Bypass (Obfuscated Payload) | **70%** | **60%** | **50%** |

***6.1: Weakness***

 **Higher Detection Rate Compared to Cobalt Strike** – Although Sliver uses **mTLS, DNS, and HTTP/S** for stealth, modern EDR/AV solutions are catching up, making **payload detection (30-70%)** higher than well-crafted Cobalt Strike beacons.

 **Limited Offensive Plugin Ecosystem** – Unlike Cobalt Strike, which has a mature market of **third-party plugins**, Sliver's **built-in exploit modules are fewer** and require more manual customization.

 **Less Polished UI & Usability** – As an **open-source project**, Sliver lacks a **fully graphical user interface (GUI)** and relies on a **command-line interface (CLI)**, making it **less user-friendly** for beginners.

 **Slower Command Execution Under Load** – While command latency is typically **150-300ms**, in high-load environments (500+ sessions), **delays can reach 500ms-1s**, affecting real-time responsiveness.

 **Lack of Native Windows Support** – Sliver is designed for **Linux and macOS**, requiring **workarounds or virtual machines** to run efficiently on **Windows servers**.

 **No Built-in Ransomware Simulation** – Unlike some C2 frameworks, Sliver does not have **dedicated ransomware modules**, limiting its use for **ransomware-specific security testing**.

***6.2: Ethical Issues***

As seen before, Sliver can be used for both legitimate and malicious activities.

* **Potential for Misuse by Threat Actors** – Since Sliver is **open-source**, cybercriminals can modify and use it for unauthorized attacks, ransomware campaigns, and persistent threats, making it a potential weapon in the wrong hands.
* **Bypassing Security Controls** – Sliver’s ability to evade antivirus (AV), endpoint detection and response (EDR), and firewalls raises concerns about its impact if misused by insiders or unethical penetration testers.
* **Privacy Violations** – Features such as keylogging, credential dumping, and remote code execution can be exploited to spy on users, steal sensitive data, and compromise privacy, violating ethical hacking guidelines.
* **Legal and Compliance Risks** – Unauthorized use of Sliver for penetration testing without explicit consent violates laws such as the Computer Fraud and Abuse Act (CFAA) and GDPR, exposing users to legal action.
* **Blurred Lines Between Ethical Hacking & Cybercrime** – Unlike proprietary tools like Cobalt Strike, which requires a license, Sliver’s free availability makes it harder to track and regulate, leading to concerns about responsible disclosure and controlled usage.

***6.3: Implementations and Adoption Challenges***

**1. Implementation Challenges**

* **Steep Learning Curve** – Lacks a GUI, requiring CLI proficiency.
* **Limited Documentation** – Fewer tutorials and troubleshooting resources.
* **Complex Setup** – Requires manual configuration of C2 channels.
* **No Native Windows Support** – Runs best on Linux/macOS, needing workarounds for Windows.
* **Frequent Updates** – Ongoing development can break existing setups.

**2. Adoption Challenges**

* **Compliance Restrictions** – Many organizations prohibit open-source C2 tools.
* **Higher Detection Rates** – AV/EDR solutions are catching up.
* **No Commercial Support** – Lacks enterprise-level assistance.
* **Perception Issues** – Viewed as a hacker tool, raising ethical concerns.

***7: Conclusion and Future Advancements***

Sliver C2 has emerged as a powerful, open-source alternative to traditional C2 frameworks like Cobalt Strike, offering stealth, flexibility, and strong adversary simulation capabilities. Its low resource footprint, multiple communication channels (mTLS, HTTP/S, DNS), and in-memory execution make it a valuable tool for red teams and penetration testers. However, higher detection rates, lack of a GUI, and adoption barriers remain challenges.

**Future Advancements**

* **Enhanced Evasion Techniques** – Further obfuscation methods to bypass modern EDR/AV solutions.
* **Better Windows Integration** – Native Windows support for easier deployment.
* **Improved User Experience** – A possible **GUI interface** for better accessibility.
* **Stronger Community & Documentation** – More tutorials, automation scripts, and community contributions.
* **AI-Powered Operations** – Leveraging machine learning for automated decision-makin**g** in attack simulations.

*References :*

* <https://darktrace.com/blog/sliver-c2-how-darktrace-provided-a-sliver-of-hope-in-the-face-of-an-emerging-c2-framework>
* <https://sliver.sh/docs?name=Getting+Started>
* <https://www.linkedin.com/pulse/sliver-c2-implementation-hands-on-approach-adegbola-adeleye-0gl6e/>
* <https://corelight.com/blog/new-sliver-c2-detection-released-redteam-detected>