DIY Malware Emulation

Build It, Break It, Detect It

DEF CON 33

#whoami



- Security Architect for the Detection and Response Team
- Design and evaluate technology for practices such as detection engineering, threat intelligence, among others
- Research

Ricardo Sanchez

@rrenatosb

#whoami



Not an expert — just someone who keeps asking “why?”

I break things, I get stuck, and I teach what helped me get unstuck.

Sebastian Tapia

@stapiadlt

Agenda



1. The problem
2. Adversary emulation
3. Choosing the right adversary
4. Emulating the adversary’s malware
5. Key takeaways

The problem

- Security tools are often deployed to “check a box”, not to stop real threats—misconfigurations and default settings are common.
- Most security validation relies on generic test cases or compliance-driven scenarios—not realistic attacker behavior.
- Defenders have limited visibility into how their controls perform against real-world adversary TTPs.

The problem

False sense of security

Adversary emulation

Cybersecurity assessment method that replicates the specific tactics, techniques, and procedures (TTPs) of real-world adversaries. It helps organizations test their defenses, uncover detection gaps, and improve response by recreating realistic attack scenarios.



Adversary emulation

Emulates real-world adversaries using their known TTPs.

Tests people, processes, and technology—not just tools.

Generates actionable insights to prioritize defensive improvements.

Choosing the right adversary





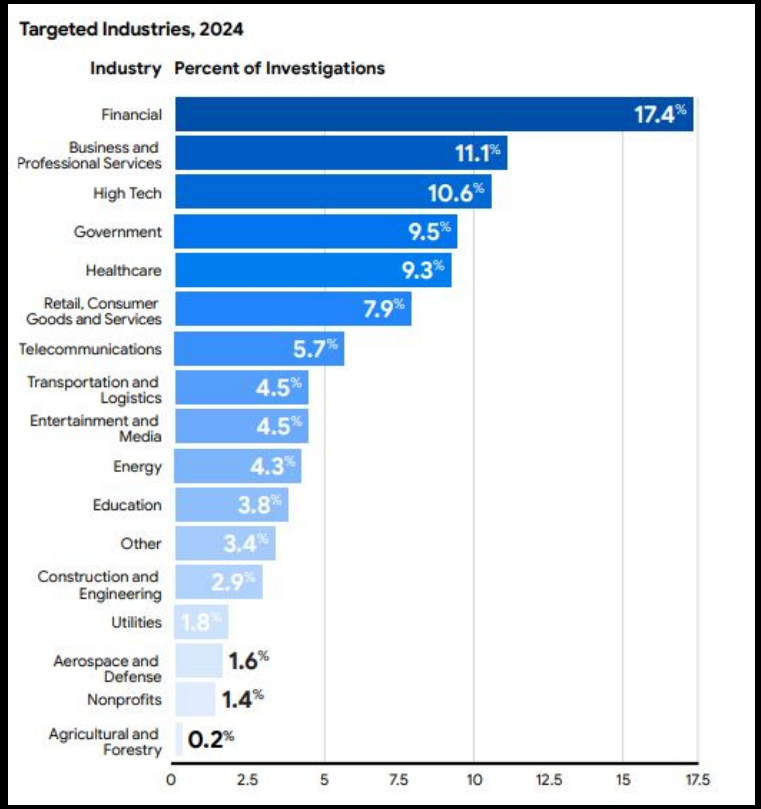


Who is attacking your region and industry?

Which TTPs do the attackers use?

Do the TTPs make sense for your environment?

Choosing the right adversary



Stolen credentials overtook email phishing as the second most frequently observed initial infection vector in 2024, representing 16% of intrusions, compared to 14% for email phishing.

Infostealer Malware Continues to Create a Threat to Enterprise Systems

In the past several years, Mandiant has seen increased attention on a specific category of malware known as info-stealers and their role in enabling often short-lived, yet deeply impactful intrusions using stolen credentials. Although infostealers and stolen credentials have always been a serious concern in cybersecurity, the recently renewed focus on infostealers by malicious actors and—consequently cybersecurity organizations—could signal drastic shifts in the ways cyber criminals abuse and/or monetize data obtained from infostealers.

Specifically, Mandiant has observed a resurgence in the use of stolen credentials as a means of initial access for compromises. While the use of stolen credentials by threat actors had dropped from 14% in 2022 to 10% in 2023, Mandiant identified stolen credentials in 16% of the intrusions observed in 2024. This resurgence is likely fueled, at least in part, by the large tranches of stolen credentials offered within cyber crime communities that have facilitated this rise in demand by offering stolen credentials in large tranches and on an individual basis.

Infostealers and broader credential theft are not new threats, but they are seeing a resurgence and have always posed significant risks to organizations that may not realize employee credentials have been compromised and exposed—sometimes years prior.

https://services.google.com/fh/files/misc/m-trends-2025-en.pdf

Choosing the right adversary

Infostealer Infections Increasingly Target Personal Devices and Obtain More Credentials per Infection, Increasing the Risk to SaaS Ecosystems

In both the Change and the Snowflake data breaches, the initial valid credentials were obtained through infostealer infections. Recorded Future observed that most infostealer infections affected personal or small-to-medium business-owned devices in 2024. Personal devices are not usually subject to the same monitoring, usage restrictions, or security resources as enterprise devices, which makes it more likely that an infostealer will not be interrupted. In addition, credentials can remain exposed long after the initial infostealer infection, as demonstrated by Mandiant's finding that some of the credentials used in the Snowflake breach were stolen in 2020.

The number of credentials stolen per device has also steadily increased since 2021, likely due to the growing number of applications users routinely log in to, both in their corporate environments and personal devices. This increases the likelihood that at least one of those credentials provides access to a corporate resource. The growing number of log-ins across devices can be difficult for security teams to manage, especially if users are logging in outside the enterprise's centralized identity management system.



https://go.recordedfuture.com/hubfs/reports/cta-2025-0128.pdf

Choosing the right adversary

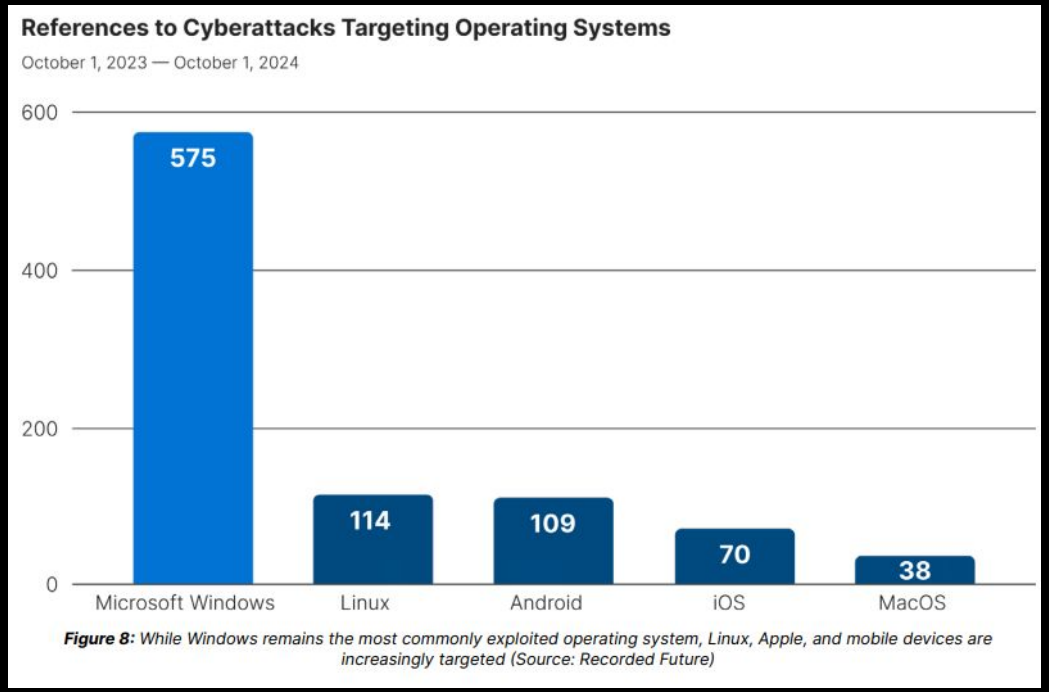
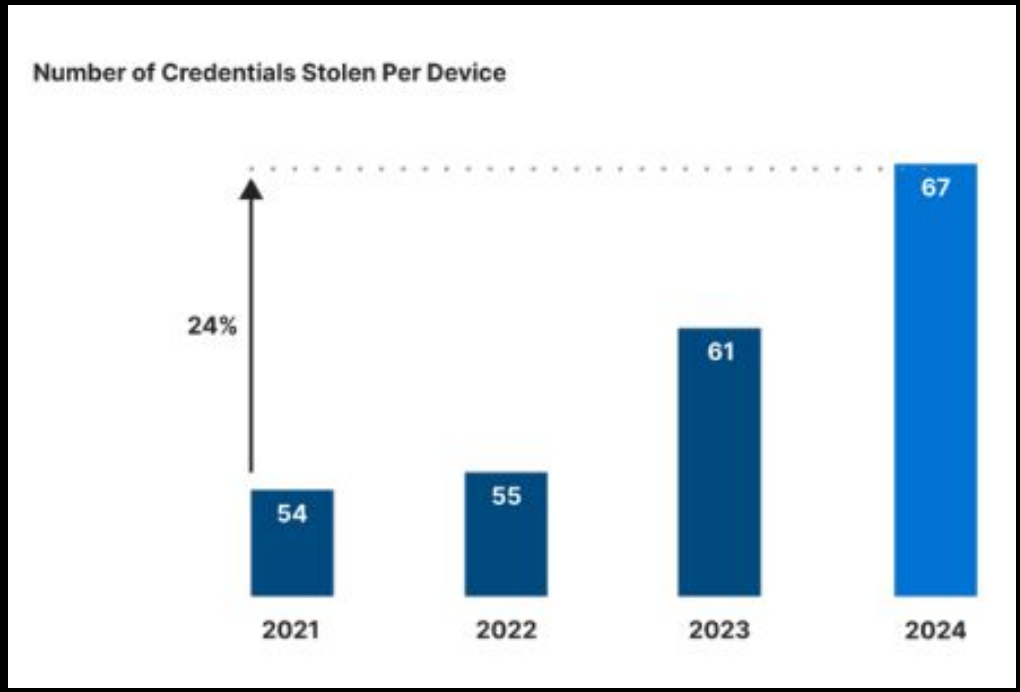
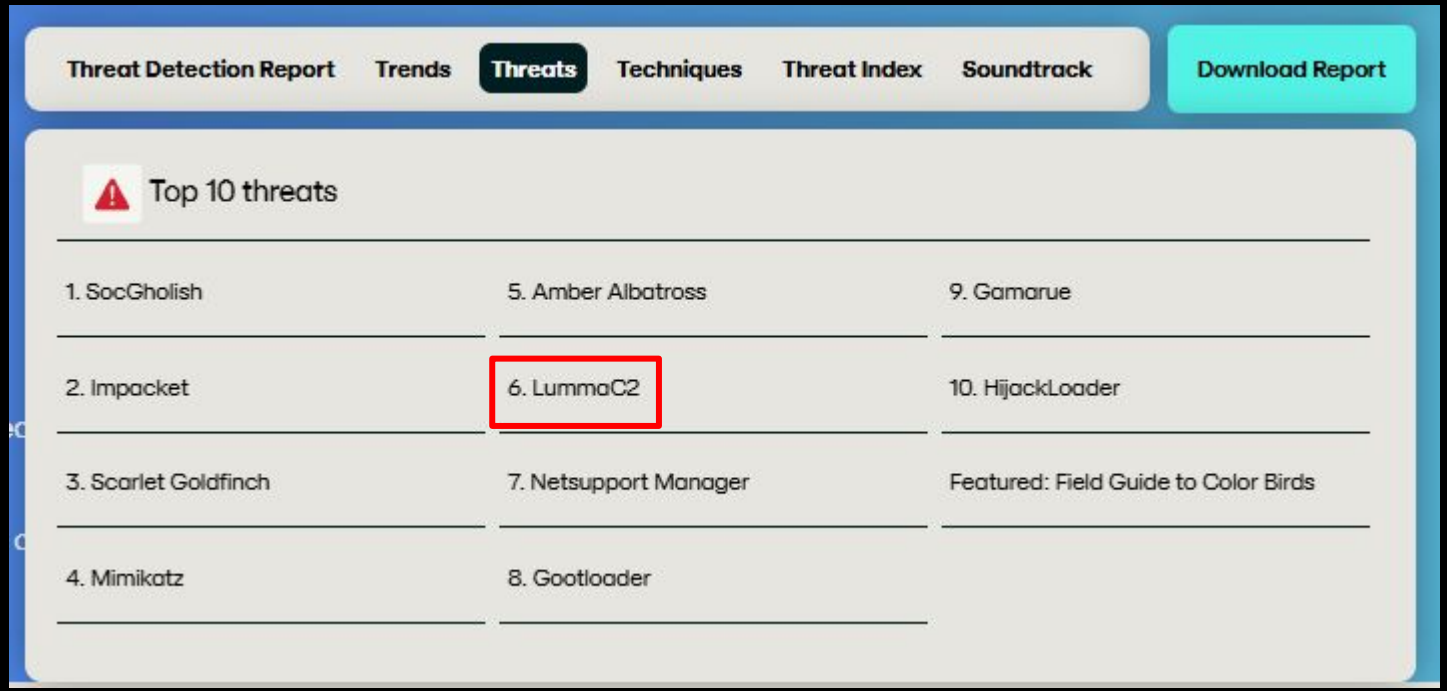


Figure 8: *While Windows remains the most commonly exploited operating system, Linux, Apple, and mobile devices are increasingly targeted (Source: Recorded Future)*



https://go.recordedfuture.com/hubfs/reports/cta-2025-0128.pdf

Choosing the right adversary



https://redcanary.com/threat-detection-report/threats/lummac2

The company

Crypket is a European fintech startup that offers cryptocurrency wallet services, crypto-to-fiat exchange, and decentralized finance (DeFi) portfolio management for retail investors.



The company

- Sector: Fintech / Cryptocurrency
- Headquarters: Barcelona, Spain
- Size: ~200 employees

Crypket

The tech stack:

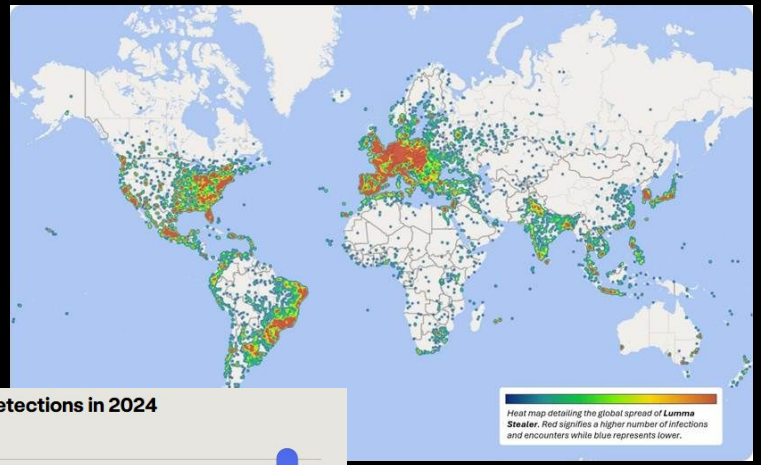
- Cloud-first architecture (Azure)
- Windows 10 corporate endpoints
- Chrome as the default browser

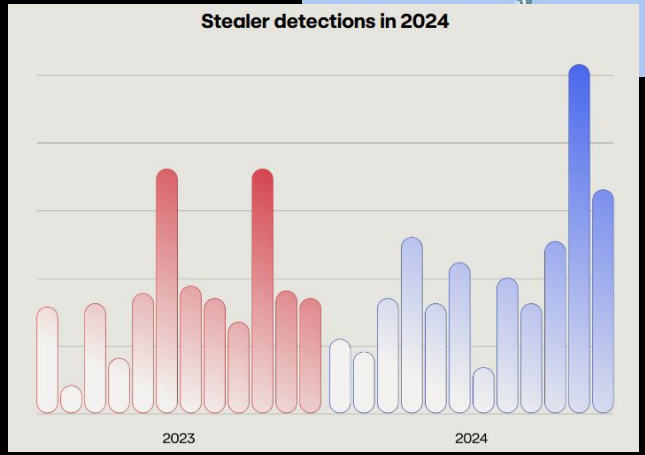




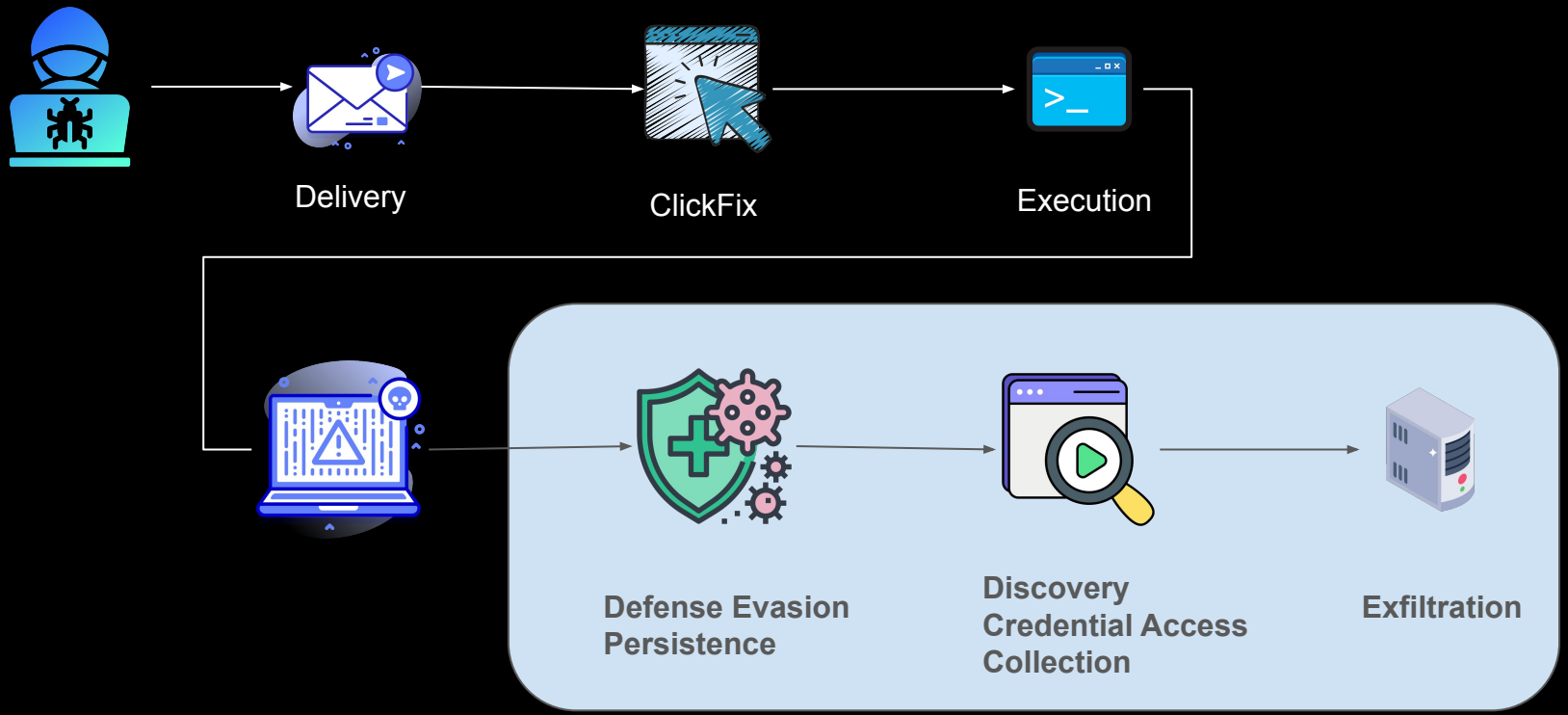
Lumma Stealer

- Also known as *LummaC2 Stealer*.
- Active since *2022*.
- Primary goal is to *steal sensitive data*.
- Operates under a *Malware-as-a-Service model*.





Lumma TTP’s


Delivery
ClickFix
Execution
Defense Evasion
Persistence
Discovery
Credential Access
Collection
Exfiltration

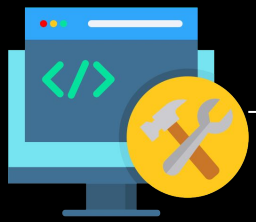
TTP’s we will emulate

Persistence	Defense Evasion	Credential Access	Discovery	Collection	Exfiltration
Modify Registry	Obfuscated Files or Information: Dynamic API Resolution	Credentials from Password Stores: Credentials from Web Browsers	System Information Discovery	Automated Collection	Automated Exfiltration
	Process Injection: Process Hollowing		Browser Information Discovery / File and Directory Discovery	Data from the Local System	
	Sandbox Evasion				
	Debugger Evasion				
	Impair Defenses: Disable Windows Event Logging				

Handling malware safely

Handling malware has real risks.

Always analyze or develop samples in a controlled, isolated environment—never on your host or production systems.

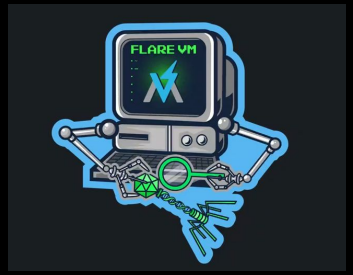


Host

VMware Workstation

Flare VM packages we will use:

- Visual Studio
- PEStudio
- x64db
- Sysinternals



Emulating Lumma

Emulating Lumma

The objective

Lumma is a stealer that targets sensitive information to be sold on criminal markets. For our adversary emulation plan, we will build a stealer that:

- Gathers credentials from browsers
- Gathers sensitive files
- Sends the information to a server

We will then check which Indicators of Compromise (IoCs) we can find in our sample, implement techniques to make analysis harder, and repeat the process.

Defense Evasion: API Hashing

Technique used by malware to **conceal the names of Windows API functions** it uses, making reverse engineering and detection harder.

How Does It Work?
📃 Compile-Time:
The function name (e.g. “VirtuallAloc”) gets hashed and hardcoded into the code

🕒 Run-Time:
1. The malware enumerates loaded DLLs and exported functions
2. Applies the same hash algorithm to each export
3. **Compares hash** to find match
4. Uses the resolved address for calling the API

Defense Evasion: Process Hollowing

Technique used by Lumma to evade detection by injecting malicious code into a legitimate process making it harder for security tools to detect the malicious activity.

How does it work?

🕒 Run-Time:

1. The malware creates a new process in a suspended state using a legitimate executable.
2. Unmaps the legitimate process's memory.
3. Writes its malicious payload into the hollowed-out memory space of the legitimate process.
4. Modifies the process's entry point or thread context.
5. Malicious code runs within the context of the legitimate process.

Defense Evasion: Debugger Evasion

Adversaries may employ various means to detect and avoid debuggers. Debuggers are typically used by defenders to trace and/or analyze the execution of potential malware payloads.

Some techniques:

- Check the Process Environment Block (PEB) to see if the BeingDebugged flag is active
- Call IsDebuggerPresent() or CheckRemoteDebuggerPresent()
- Call GetTickCount() to detect delays caused by breakpoints

https://attack.mitre.org/

Defense Evasion: Sandboxing Evasion

Malware authors design sandboxing evasion techniques to ensure their code behaves benignly or remains dormant in a sandbox, preventing analysts from uncovering its malicious capabilities.

Some techniques:

- Check for virtualization artifacts, such as VMware, VirtualBox (e.g., VBoxMouse.sys).
- Detect sandbox-specific processes or services (e.g., SbieCtrl.exe for Sandboxie).
- Look for analysis tools like Wireshark, Process Explorer, or debuggers (e.g., OllyDbg, IDA Pro).

Defense Evasion: Disable Windows Event Logging

Malware uses this technique to stop, clear, or manipulate the Windows Event Log, a system component that records system, application, and security events on Windows operating systems making it harder for security tools and analysts to detect, investigate, or respond to malicious behavior.

Some techniques:

- Terminate the Windows Event Log service (eventlog) using commands like net stop eventlog.
- Modify registry keys (HKLM\SYSTEM\CurrentControlSet\Services\EventLog) to disable logging or restrict access to the Event Log service.
- Modify Group Policy or audit policies to disable logging for specific events, such as process creation (Event ID 4688) or account logons (Event ID 4624).

Persistence: Registry Run Keys

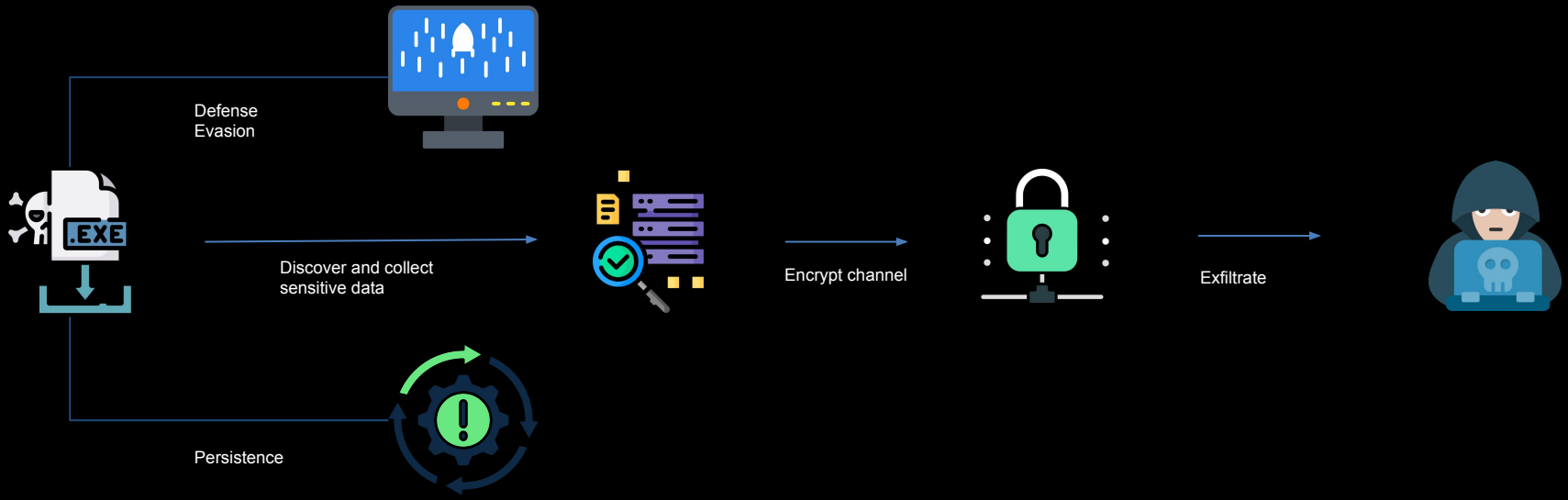
Adding an entry to the "run keys" in the Registry or startup folder will cause the program referenced to be executed when a user logs in. These programs will be executed under the context of the user and will have the account's associated permissions level.

The following run keys are created by default on Windows systems:

- HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
- HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce
- HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run
- HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce

https://attack.mitre.org/

Looking at the kill chain



Next steps

1. Test in a Controlled Environment
2. Document the Emulation Plan
3. Notify Key Stakeholders
4. Capture Detection & Response Metrics
5. Review, Tune, and Re-Test

Key Takeaways

1. **You can’t emulate what you don’t understand**
Threat intelligence is your foundation—know your adversary before you simulate them.

2. **Context is critical**
Choose adversaries that align with your sector, architecture, and threat landscape.

3. **Continuous testing is the goal**
Emulation isn't a one-time event—it should drive ongoing validation and improvement.

4. **Precision beats complexity**
Effective emulation isn’t about doing everything—focus your emulation on what matters.

THANK YOU!

DIY Malware Emulation

Build It, Break It, Detect It

@rrenatosb

@stapiadlt