MALPROXYING: LEAVE YOUR MALWARE AT HOME

By Amit Waisel, Senior Technology Lead in Security Research, XM Cyber

Endpoint protection plays a critical role in the modern organizational security stack. Yet the very nature of this security model is fundamentally flawed. Endpoint security solutions, and the malicious actors trying to breach them, are locked into a perpetual game of cat and mouse. Each side must continually adapt and react to the tactics of the other. And, unfortunately for organizational security specialists, the playing field is radically unbalanced.

Security solutions and professionals need to maintain perfect endpoint protection; hackers, meanwhile, need only a single successful attempt to wreak extraordinary damage. Yet security solutions do have one point in their favor: The most common endpoint security evasion techniques require constant updating which limits the pool of attackers and the scale at which attacks are launched.

This leads to a troubling question -- what if a technique existed that allowed attackers to evade defense mechanisms while requiring little in the way of adjustments to malicious code? That was the topic of a well-received recent presentation I gave along with my colleague security researcher Hila Cohen at DEF CON 27 in Las Vegas, Nevada.

Let's take a closer look at this technique and its implications for endpoint security.

THE CURRENT STATE OF ENDPOINT SECURITY

Existing security solutions use three mechanisms to maintain protection:

- **Static signatures** -- these can be a simple hash from a sequence of bytes in a file. Signatures sign file segments (or memory blocks), enabling a check against common IOCs (Indicators of Compromise) to see if the file is infected.
- **Heuristic rules** -- these rules can inspect the imported function list, executable uses, its sections sizes and structure, and many more properties including entropy. Heuristic rules attempt to discern properties that are common among malicious files yet don't exist in safe executables. They are not based on IOCs and don't examine binary sequences or hashes included in the static signature category.
- **Behavioral signatures** -- these signatures attempt to identify, evaluate and block all malicious activity. Because of the limitations of static signatures and heuristic rules, infected files are often miscategorized as safe. Behavioral signatures take a different approach, as they are based on an operational sequence executed in the system, rather than the implementation of malicious logic.

As mentioned above, endpoint protection solutions have a variety of weaknesses. Attackers can change the IOCs, properties and behavior of malicious files, allowing them to evade detection and quarantining. However, these techniques are highly manual and require significant expertise, making it difficult for attackers to implement at scale.

There is, however, another approach enabling the circumvention of endpoint security without the need for extensive labor or expertise: Malproxying.

HOW MALPROXYING WORKS

The core operational model of endpoint security solutions is simple: Identify and analyze code, then classify and (potentially) block. Yet what if an attacker could obscure that code entirely?

That's the premise of the malproxying technique, which avoids deploying malicious code on target machines and therefore separates that code from any interaction with the target operating system. Here's how it works:

A piece of code interacts with its operating system and environment through a set of API calls. The attacker redirects those API calls, and instead of running them on his operating system, he proxies them over the network to the target machine. So, the malicious code resides on the attacker side, where it is not monitored by any security solution (as the attacker completely controls the environment), but the actions performed by that malicious code actually interact with the target environment, allowing it to bypass common endpoint security protection mechanisms. The malicious code, meanwhile, cannot tell that it has not been executed on the targeted machine.

On a deeper level, the technique involves two key components: attacker and target stubs. The attacker code loads and executes malicious instructions, controls its API function calls and redirects them over a network tunnel to the target stub.

The target code appears innocent and has no malicious activity pre-coded. It receives the API requests and parameters, executes those requests and returns the results back to the attacker stub. These results are returned to the malicious code, in the exact way they would be returned if the malicious code had called the API functions locally. The malicious code is totally unaware of the long journey the response went through until it arrived at its destination.

COUNTERING MALPROXYING

The malproxying technique is designed to evade the primary mechanisms used by endpoint detection solutions. The target stub contains no malicious logic in its base form, rendering it hard to identify and easy to modify if caught. Static signatures and heuristic rules are easily bypassed.

Behavioral signatures, however, are another matter. In the bottom line, a “malicious” sequence of API calls must be executed on the target machine to achieve the attacker’s malicious goals. A sophisticated monitoring tool can detect that malicious flow and trigger an alarm. This merely invites another protracted cat and mouse battle, as the attackers have to find new ways to make it very hard for monitoring tools to assemble the trace of their malicious actions.

For example, an attacker could trigger each API function call in a different thread, making it harder for security solutions to identify a single code flow to check whether it is malicious or not. Second, the attacker could bypass the detection points, where the security solution tracks the activity of our process. Once those detection points are bypassed, the security solution is blind to any API-based activity.

Continual improvement and refinement of behavioral detection capabilities represent a better option. Actions triggered by malicious logic can be tracked using various techniques to ensure that calls are fully tracked. By building a more robust log of executed system function calls -- and the signatures that define malicious behavior -- organizations can develop a more viable line of defense against this novel attack technique.

"On a deeper level, the technique involves two key components:

Attacker and target stubs.

The attacker code loads and executes malicious instructions, controls its API function calls and redirects them over a network tunnel to the target stub."

ABOUT XM CYBER

XM Cyber was founded by security executives from the elite Israeli intelligence sector.

XM Cyber's core team is comprised of highly skilled and experienced veterans from the Israeli Intelligence with expertise in both offensive and defensive cyber security.

XM Cyber has developed more than 15 patented technologies based on a proprietary set of algorithms that enable the continuous and automatic simulation of a hacker's techniques and methods.

AMIT WAISEL

Amit Waisel is a Senior Technology lead in Security Research at XM Cyber. He is a seasoned data security expert with vast experience in cyber offensive projects. Prior to XM Cyber, Waisel filled multiple data security positions in the Israeli intelligence community. Waisel is well experienced with malware detection and analysis techniques, operating system internals and security-oriented software development.