**Analysis of a Malicious Microsoft-Signed Driver**

Recently, there has been a trend of malicious kernel mode drivers being signed by Microsoft, which poses a significant security threat. A recent Microsoft-signed driver of interest belongs to a small toolkit designed to terminate AV and EDR processes. This is not the first time a malicious kernel mode driver has been signed by Microsoft. In June 2021, GData published a report on a similar activity. The main issue with this process is that most security solutions implicitly trust anything signed by only Microsoft, thereby subverting attempts to detect or root them out. That battle has been going on for quite some time.

**Background on the Driver Signing Process**

To be加载 onto the system, a driver must have a valid cryptographic signature. This signature is used to verify the identity of the developer and their code. The developer must first obtain an Extended Validation (EV) certificate and attach it to their account after registering for the WHQL signing process. The drivers are run through a Partner Portal for testing and signature. The drivers are run on a system that simulates the environment in which they will be installed. The ability to simulate such an environment is important in combating the scourge of kernel mode rootkits, malware designed to run with the highest privileges and capabilities.

**Microsoft's “Windows Hardware Compatibility Publisher” certificate**

In the deployment of various industry verticals, such as Pharmaceuticals, Healthcare, Financial, and Cryptocurrency, the use of signed drivers has increased. In 2022, the actors were involved in a variety of intrusions heavily targeting Business Process Outsourcing (BPO). Detailed investigations led to the discovery of POORTRY and STONESTOP malware, part of a small toolkit we observed in different time periods.

**Versioning and Signing Considerations**

A recent Microsoft-signed driver of interest belongs to a small toolkit designed to terminate AV and EDR processes. The first version of POORTRY was not signed directly by Microsoft. The second version was VMProtected and signed through the WHQL signing process. The third version was also signed through the WHQL signing process, but was protected with an unknown protection method.

We observed three versions of the driver toolkit that the threat actors have used in separate time periods. The first version of POORTRY was not packed. The second version was packed using VMProtect. The POORTRY variant of the second version of the toolkit we observed was packed using VMProtect. The STONESTOP variant of this toolkit does not feature an external configuration file but embeds target process names.

- **Detection Opportunity**

POORTY offers functionality to delete files from disk, as well as overwrite files already present on disk. Additional functionality in the third version of the toolkit includes the ability to tamper with files. The file-tampering functionality is identical to version 2, version 3 does not contain the PID masking feature/ protection mechanism. It also does not feature an external configuration file.

**Details**

Before reading from the file, STONESTOP verifies the file's integrity against a predefined MD5 hash and terminates if the hash value does not match. If leveraged against the two malicious drivers in our instance, we are able to extract the original requesting process ID (PID). The STONESTOP variant of this toolkit does not feature an external configuration file but embeds target process names.

**How STONESTOP handles PIDs**

When reading from the configuration file, STONESTOP determines a DWORD value. POORTY then responds with a corresponding token. This process is repeated until all PIDs are determined. STONESTOP masks PIDs using a value derived from the MAC address of the disk it is executing on. The target PIDs are sent to POORTY. STONESTOP masks PIDs using a value derived from a SHA-256 hash of a plaintext string.

**Signature Verification**

In the meantime, we continue to monitor the abuse of signed drivers in collaboration with Mandiant researchers. We are continuing to improve enhancements to bolster the security of their signing process to help maintain the implicit trust placed in Microsoft signing. The importance of driver signing mechanisms is evident, especially in modern operating systems. The introduction of driver signing with the WHQL process was key in stemming the tide of rootkits for years. The receding effectiveness of code signing represents a significant concern for security professionals.

**Future Research Opportunities**

This research is being released alongside Mandiant, a SentinelOne technology and incident response partner. We are continuing to monitor the abuse of signed drivers and are working with Mandiant researchers to improve enhancements to bolster the security of their signing process to help maintain the implicit trust placed in Microsoft signing.