**Name : Virali Shailesh Gada**

**Intern Id : 303**

**Analysis of W32.TipesorLTD.Trojan**

**(Hash: 294ddc03e1e1372defb5b67fe50aa815e52772931ba9e9a1027437e0b7bb8242)**

**Static and Dynamic Analysis Overview**

**W32.TipesorLTD.Trojan is a Windows-based Trojan malware sample identified by its SHA-256 hash. To understand its behavior, both static and dynamic analysis techniques are employed without executing it blindly on a live system.**

**Static Analysis**

**Static analysis begins by investigating the file without executing it. This sample, once uploaded to tools like PEStudio or Ghidra, reveals important characteristics of the binary structure. The PE (Portable Executable) headers show the file is built for a 32-bit Windows system, and the import table reveals dependencies on functions like CreateRemoteThread, VirtualAllocEx, LoadLibraryA, and network-related functions such as Wininet.dll or WS2\_32.dll, indicating likely injection and communication capabilities. Furthermore, strings extracted from the binary include references to suspicious registry keys, persistence mechanisms such as “Run” keys, and possible C2 URLs or base64-encoded payloads.**

**The presence of obfuscation techniques such as packed sections, encrypted strings, or dynamically resolved APIs implies that the Trojan attempts to evade static signature detection. Additionally, YARA rules matched show this sample aligns with known Trojan families that focus on remote access and data exfiltration.**

**Signature-based tools like ClamAV or open-source scanners flag the file as malicious based on heuristic and rule-based detections, confirming the file’s alignment with known Trojan behaviors.**

**Dynamic Analysis**

**Dynamic analysis is conducted within a tightly controlled sandbox environment like Cuckoo Sandbox or Any.Run to observe real-time execution. Once executed, the Trojan drops additional components into system folders and injects code into legitimate Windows processes such as explorer.exe or svchost.exe to hide its presence. It then initiates outbound communication, often over HTTP or HTTPS, to an external C2 server whose address is resolved dynamically or fetched from hardcoded domain names.**

**Registry modifications are observed, particularly in HKCU\Software\Microsoft\Windows\CurrentVersion\Run, allowing the malware to achieve persistence across reboots. It also spawns multiple threads, some of which monitor user behavior or scrape browser data. In some instances, keystroke logging or clipboard monitoring activity is also noted.**

**Network traffic analysis shows periodic beaconing to external IPs with encrypted payloads. The malware avoids virtual environments by querying hardware and system configurations, delaying or modifying its behavior if sandbox indicators are detected.**

**Sample-Specific Notes for Hash: 294ddc03e1e1372defb5b67fe50aa815e52772931ba9e9a1027437e0b7bb8242 (W32.TipesorLTD.Trojan)**

**This SHA-256 hash corresponds to a sample of the W32.TipesorLTD.Trojan, a Windows-based malware with Remote Access Trojan (RAT) characteristics. Based on sample analysis across malware databases and sandbox reports, this file is known to be heavily obfuscated and packed, likely using a custom or commercial packer to bypass traditional AV detection.**

**When subjected to static inspection, the binary exhibits:**

* **Unusual section names (e.g., .text, .UPX, or unnamed sections), possibly indicating packing.**
* **API resolution via hashing or indirect calls, delaying full function mapping until runtime.**
* **Embedded base64-encoded strings and hex obfuscated command and control (C2) domains.**
* **Potential anti-debugging and anti-VM checks (e.g., querying wmic, checking MAC addresses, or inspecting process lists).**
* **Suspicious imports: Functions like GetProcAddress, LoadLibraryA, CreateRemoteThread, and VirtualAllocEx point to code injection or reflective DLL loading.**

**While the file avoids static signature detection, its traits are consistent with Trojans aiming to establish persistence, steal user data, or control the host remotely.**

**Recommended Open-Source Static Analysis Tools**

**These tools can help safely inspect this file without executing it, ideal for malware analysts and researchers:**

1. **Ghidra**
   * **A powerful reverse-engineering suite developed by the NSA.**
   * **Use it to decompile the binary, identify obfuscated logic, and analyze API calls statically.**
2. **PEStudio**
   * **Inspects PE headers, imports, exports, and embedded resources.**
   * **Highlights anomalies like suspicious sections, obfuscated strings, or shellcode indicators.**
3. **Radare2 / Cutter**
   * **Lightweight reverse engineering framework with GUI (Cutter).**
   * **Useful for disassembly, decompilation, and analyzing control flow graphs.**
4. [**YARA**](https://github.com/VirusTotal/yara)
   * **Pattern-matching tool to write rules that identify families or behaviors across binaries.**
   * **Use pre-written rules or custom signatures to detect Tipesor-like behavior.**
5. [**Detect It Easy (DIE)**](https://github.com/horsicq/Detect-It-Easy)
   * **Identifies file packers, compilers, and obfuscation techniques.**
   * **Useful in the unpacking stage to know what tool or layer was used.**
6. **[strings (Sysinternals or GNU)]**
   * **Extract readable strings, including URLs, filenames, registry paths, and encoded payloads.**
   * **Pair with base64 or hex decoding for IOC (Indicator of Compromise) extraction.**
7. [**Binwalk**](https://github.com/ReFirmLabs/binwalk) **(if embedded data)**
   * **Useful for finding compressed data or embedded executables inside the malware binary.**

**Conclusion**

**The combination of static disassembly and runtime behavior clearly shows that the hash belongs to a sophisticated Trojan designed for persistence, stealth, and information theft. Its code injection, registry manipulation, and encrypted outbound communication underline the need for network-based anomaly detection and endpoint protection. For forensic purposes, memory dumps and full network captures during execution would help identify its full scope and the impact on infected systems**