Jul 11 2024

NETMANAGEIT

Intelligence Report

DodgeBox: A deep dive into the updated arsenal of APT41

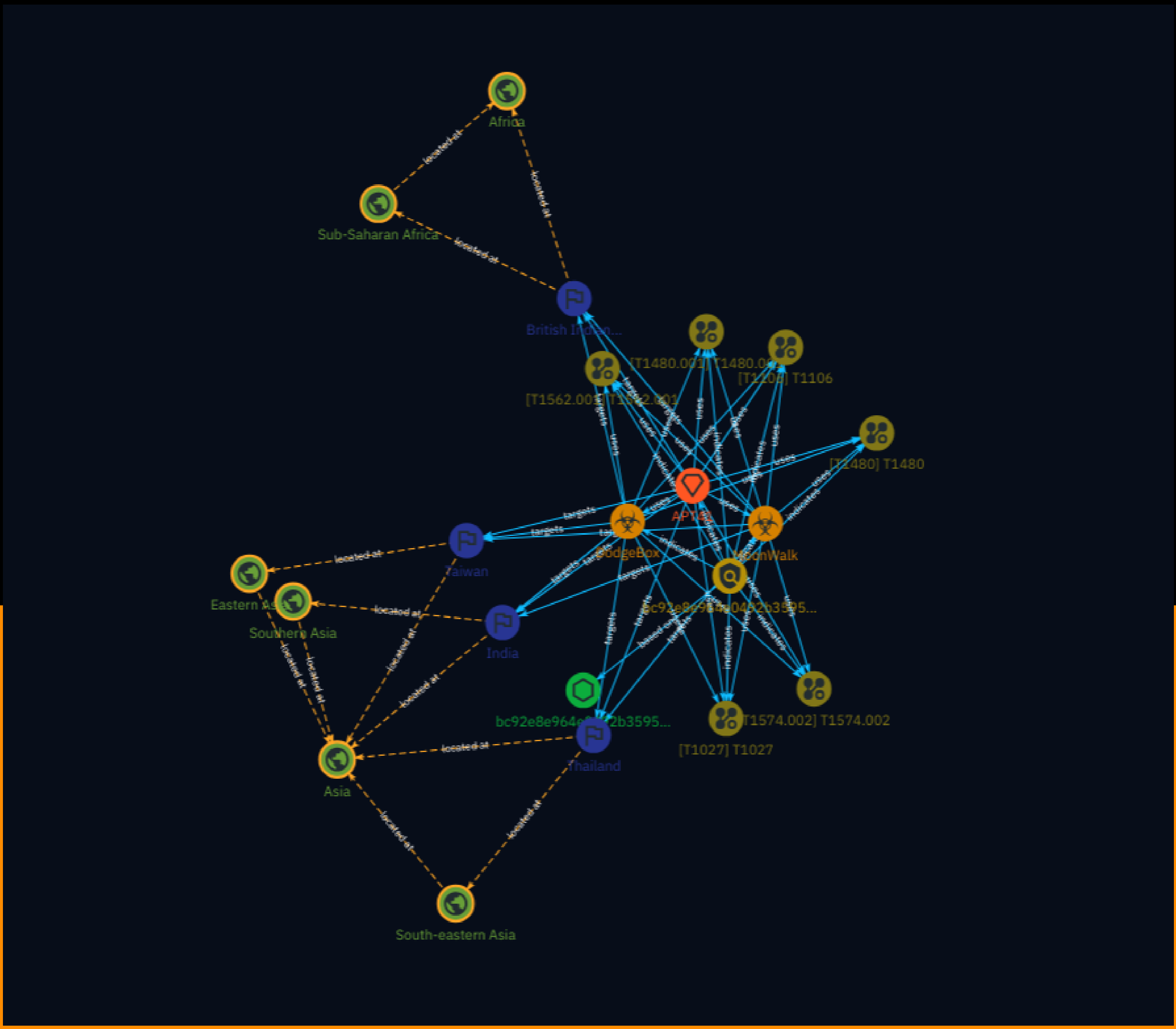

Table of contents

Overview

Entities

Observables

External References

Overview

Description

This blog post provides an in-depth technical analysis of a newly discovered malware loader called DodgeBox, which is attributed to the China-based advanced persistent threat (APT) actor APT41. DodgeBox incorporates various evasion techniques such as call stack spoofing, DLL sideloading, DLL hollowing, and environmental guardrails to evade detection. The analysis also highlights the similarities between DodgeBox and the previously known StealthVector tool associated with APT41, leading to the attribution of this new malware to the same threat actor with moderate confidence.

Confidence

This value represents the confidence in the correctness of the data contained within this report.

100 / 100

Content

N/A

Attack-Pattern

Name

T1106

ID

T1106

Description

Adversaries may interact with the native OS application programming interface (API) to execute behaviors. Native APIs provide a controlled means of calling low-level OS services within the kernel, such as those involving hardware/devices, memory, and processes.(Citation: NT API Windows)(Citation: Linux Kernel API) These native APIs are leveraged by the OS during system boot (when other system components are not yet initialized) as well as carrying out tasks and requests during routine operations. Adversaries may abuse these OS API functions as a means of executing behaviors. Similar to [Command and Scripting Interpreter](https://attack.mitre.org/techniques/T1059), the native API and its hierarchy of interfaces provide mechanisms to interact with and utilize various components of a victimized system. Native API functions (such as `NtCreateProcess`) may be directed invoked via system calls / syscalls, but these features are also often exposed to user-mode applications via interfaces and libraries.(Citation: OutFlank System Calls)(Citation: CyberBit System Calls)(Citation: MDSec System Calls) For example, functions such as the Windows API `CreateProcess()` or GNU `fork()` will allow programs and scripts to start other processes.(Citation: Microsoft CreateProcess)(Citation: GNU Fork) This may allow API callers to execute a binary, run a CLI command, load modules, etc. as thousands of similar API functions exist for various system operations.(Citation: Microsoft Win32)(Citation: LIBC)(Citation: GLIBC) Higher level software frameworks, such as Microsoft .NET and macOS Cocoa, are also available to interact with native APIs. These frameworks typically provide language wrappers/abstractions to API functionalities and are designed for ease-of-use/portability of code.(Citation: Microsoft NET)(Citation: Apple Core Services)(Citation: MACOS

Cocoa)(Citation: macOS Foundation) Adversaries may use assembly to directly or in-directly invoke syscalls in an attempt to subvert defensive sensors and detection signatures such as user mode API-hooks.(Citation: Redops Syscalls) Adversaries may also attempt to tamper with sensors and defensive tools associated with API monitoring, such as unhooking monitored functions via [Disable or Modify Tools](https://attack.mitre.org/techniques/T1562/001).

Name

T1027

ID

T1027

Description

Adversaries may attempt to make an executable or file difficult to discover or analyze by encrypting, encoding, or otherwise obfuscating its contents on the system or in transit. This is common behavior that can be used across different platforms and the network to evade defenses. Payloads may be compressed, archived, or encrypted in order to avoid detection. These payloads may be used during Initial Access or later to mitigate detection. Sometimes a user's action may be required to open and [Deobfuscate/Decode Files or Information](https://attack.mitre.org/techniques/T1140) for [User Execution](https://attack.mitre.org/techniques/T1204). The user may also be required to input a password to open a password protected compressed/encrypted file that was provided by the adversary. (Citation: Volexity PowerDuke November 2016) Adversaries may also use compressed or archived scripts, such as JavaScript. Portions of files can also be encoded to hide the plain-text strings that would otherwise help defenders with discovery. (Citation: Linux/Cdorked.A We Live Security Analysis) Payloads may also be split into separate, seemingly benign files that only reveal malicious functionality when reassembled. (Citation: Carbon Black Obfuscation Sept 2016) Adversaries may also abuse [Command Obfuscation](https://attack.mitre.org/techniques/T1027/010) to obscure commands executed from payloads or directly via [Command and Scripting Interpreter](https://attack.mitre.org/techniques/T1059). Environment variables, aliases, characters, and other platform/language specific semantics can be used to evade signature based detections and application control mechanisms. (Citation: FireEye Obfuscation June 2017) (Citation: FireEye Revoke-Obfuscation July 2017)(Citation: PaloAlto EncodedCommand March 2017)

Name

T1480

ID

T1480

Description

Adversaries may use execution guardrails to constrain execution or actions based on adversary supplied and environment specific conditions that are expected to be present on the target. Guardrails ensure that a payload only executes against an intended target and reduces collateral damage from an adversary’s campaign.(Citation: FireEye Kevin Mandia Guardrails) Values an adversary can provide about a target system or environment to use as guardrails may include specific network share names, attached physical devices, files, joined Active Directory (AD) domains, and local/external IP addresses.(Citation: FireEye Outlook Dec 2019) Guardrails can be used to prevent exposure of capabilities in environments that are not intended to be compromised or operated within. This use of guardrails is distinct from typical [Virtualization/Sandbox Evasion](https://attack.mitre.org/techniques/T1497). While use of [Virtualization/Sandbox Evasion](https://attack.mitre.org/techniques/T1497) may involve checking for known sandbox values and continuing with execution only if there is no match, the use of guardrails will involve checking for an expected target-specific value and only continuing with execution if there is such a match.

Name

T1480.001

ID

T1480.001

Description

Adversaries may environmentally key payloads or other features of malware to evade defenses and constraint execution to a specific target environment. Environmental keying uses cryptography to constrain execution or actions based on adversary supplied environment specific conditions that are expected to be present on the target.

Environmental keying is an implementation of [Execution Guardrails](https://attack.mitre.org/techniques/T1480) that utilizes cryptographic techniques for deriving encryption/decryption keys from specific types of values in a given computing environment.(Citation: EK Clueless Agents) Values can be derived from target-specific elements and used to generate a decryption key for an encrypted payload. Target-specific values can be derived from specific network shares, physical devices, software/software versions, files, joined AD domains, system time, and local/external IP addresses.(Citation: Kaspersky Gauss Whitepaper)(Citation: Proofpoint Router Malvertising)(Citation: EK Impeding Malware Analysis)(Citation: Environmental Keyed HTA)(Citation: Ebowla: Genetic Malware) By generating the decryption keys from target-specific environmental values, environmental keying can make sandbox detection, anti-virus detection, crowdsourcing of information, and reverse engineering difficult.(Citation: Kaspersky Gauss Whitepaper)(Citation: Ebowla: Genetic Malware) These difficulties can slow down the incident response process and help adversaries hide their tactics, techniques, and procedures (TTPs). Similar to [Obfuscated Files or Information](https://attack.mitre.org/techniques/T1027), adversaries may use environmental keying to help protect their TTPs and evade detection. Environmental keying may be used to deliver an encrypted payload to the target that will use target-specific values to decrypt the payload before execution.(Citation: Kaspersky Gauss Whitepaper)(Citation: EK Impeding Malware Analysis)(Citation: Environmental Keyed HTA)(Citation: Ebowla: Genetic Malware)(Citation: Demiguise Guardrail Router Logo) By utilizing target-specific values to decrypt the payload the adversary can avoid packaging the decryption key with the payload or sending it over a potentially monitored network connection. Depending on the technique for gathering target-specific values, reverse engineering of the encrypted payload can be exceptionally difficult.(Citation: Kaspersky Gauss Whitepaper) This can be used to prevent exposure of capabilities in environments that are not intended to be compromised or operated within. Like other [Execution Guardrails](https://attack.mitre.org/techniques/T1480), environmental keying can be used to prevent exposure of capabilities in environments that are not intended to be compromised or operated within. This activity is distinct from typical [Virtualization/Sandbox Evasion](https://attack.mitre.org/techniques/T1497). While use of [Virtualization/Sandbox Evasion](https://attack.mitre.org/techniques/T1497) may involve checking for known sandbox values and continuing with execution only if there is no match, the use of environmental keying will involve checking for an expected target-specific value that must match for decryption and subsequent execution to be successful.

Name
T1574.002

ID

T1574.002

Description

Adversaries may execute their own malicious payloads by side-loading DLLs. Similar to [DLL Search Order Hijacking](https://attack.mitre.org/techniques/T1574/001), side-loading involves hijacking which DLL a program loads. But rather than just planting the DLL within the search order of a program then waiting for the victim application to be invoked, adversaries may directly side-load their payloads by planting then invoking a legitimate application that executes their payload(s). Side-loading takes advantage of the DLL search order used by the loader by positioning both the victim application and malicious payload(s) alongside each other. Adversaries likely use side-loading as a means of masking actions they perform under a legitimate, trusted, and potentially elevated system or software process. Benign executables used to side-load payloads may not be flagged during delivery and/or execution. Adversary payloads may also be encrypted/packed or otherwise obfuscated until loaded into the memory of the trusted process.(Citation: FireEye DLL Side-Loading)

Name

T1562.001

ID

T1562.001

Description

Adversaries may modify and/or disable security tools to avoid possible detection of their malware/tools and activities. This may take many forms, such as killing security software processes or services, modifying / deleting Registry keys or configuration files so that tools do not operate properly, or other methods to interfere with security tools scanning or reporting information. Adversaries may also disable updates to prevent the latest security patches from reaching tools on victim systems.(Citation: SCADAfence_ransomware) Adversaries may also tamper with artifacts deployed and utilized by security tools. Security tools may make dynamic changes to system components in order to maintain visibility into specific events. For example, security products may load their own modules and/or modify those loaded by processes to facilitate data collection. Similar to [Indicator Blocking](https://attack.mitre.org/techniques/T1562/006), adversaries may unhook or otherwise modify these features added by tools (especially those that exist in userland or are otherwise potentially accessible to adversaries) to avoid detection.(Citation: OutFlank System Calls)(Citation: MDSec System Calls) Adversaries may also focus on specific applications such as Sysmon. For example, the “Start” and “Enable” values in `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\WMI\Autologger\EventLog-Microsoft-Windows-Sysmon-Operational` may be modified to tamper with and potentially disable Sysmon logging.(Citation: disable_win_evt_logging) On network devices, adversaries may attempt to skip digital signature verification checks by altering startup configuration files and effectively disabling firmware verification that typically occurs at boot.(Citation: Fortinet Zero-Day and Custom Malware Used by Suspected Chinese Actor in Espionage Operation)(Citation: Analysis of FG-IR-22-369) In cloud environments, tools disabled by adversaries may include cloud monitoring agents that report back to services such as AWS CloudWatch or Google Cloud Monitor. Furthermore, although defensive tools may have anti-tampering mechanisms, adversaries may abuse tools such as legitimate rootkit removal kits to impair and/or disable these tools.(Citation: chasing_avaddon_ransomware)(Citation: dharma_ransomware)(Citation: demystifying_ryuk)(Citation: doppelpaymer_crowdstrike) For example, adversaries have used tools such as GMER to find and shut down hidden processes and antivirus software on infected systems.(Citation: demystifying_ryuk) Additionally, adversaries may exploit legitimate drivers from anti-virus software to gain access to kernel space (i.e. [Exploitation for Privilege Escalation](https://attack.mitre.org/techniques/T1068)), which may lead to bypassing anti-tampering features.(Citation: avoslocker_ransomware)

Indicator

Name

bc92e8e964e0492b3595d9470e59941bded90082040ac436583b9f3269e1e550

Pattern Type

stix

Pattern

[file:hashes.'SHA-256' =
'bc92e8e964e0492b3595d9470e59941bded90082040ac436583b9f3269e1e550']

Intrusion-Set

Name
APT41

Description
[APT41](https://attack.mitre.org/groups/G0096) is a threat group that researchers have assessed as Chinese state-sponsored espionage group that also conducts financially-motivated operations. Active since at least 2012, [APT41](https://attack.mitre.org/groups/G0096) has been observed targeting healthcare, telecom, technology, and video game industries in 14 countries. [APT41](https://attack.mitre.org/groups/G0096) overlaps at least partially with public reporting on groups including BARIUM and [Winnti Group](https://attack.mitre.org/groups/G0044).(Citation: FireEye APT41 Aug 2019)(Citation: Group IB APT 41 June 2021)

Country

Name
Taiwan

Name
India

Name
British Indian Ocean Territory

Name
Thailand

Region

Name
Asia

Name
Sub-Saharan Africa

Name
Africa

Name
Southern Asia

Name
Eastern Asia

Name
South-eastern Asia

Malware

Name
DodgeBox

Name
MoonWalk

uses

Name

Description

[APT41](https://attack.mitre.org/groups/G0096) used VMProtected binaries in multiple intrusions.(Citation: FireEye APT41 March 2020)

Name

Name

Name

Name

Name

Name

Name
Name
Description
[APT41](https://attack.mitre.org/groups/G0096) used legitimate executables to perform DLL side-loading of their malware.(Citation: FireEye APT41 Aug 2019)
Name
Name
Name
Name
Name
Name
Name
Name

Name
Description
[APT41](https://attack.mitre.org/groups/G0096) has encrypted payloads using the Data Protection API (DPAPI), which relies on keys tied to specific user accounts on specific machines. [APT41](https://attack.mitre.org/groups/G0096) has also environmentally keyed second stage malware with an RC5 key derived in part from the infected system's volume serial number.(Citation: Twitter ItsReallyNick APT41 EK)
Name
Name

indicates

Name

Name

Name

Name

Name

Name

Name

Name

Name

located-at

Name

Name

Description

Country India is located in Southern Asia

Name

Description

Region Eastern Asia is located in Asia

Name

Description

Region South-eastern Asia is located in Asia

Name

Description
Region Sub-Saharan Africa is located in Africa

Name

Name

Description
Region Southern Asia is located in Asia

Name

Name

Description
Country Thailand is located in South-eastern Asia

Name

Description
Country Taiwan is located in Eastern Asia

Name

Name

Description
Country British Indian Ocean Territory is located in Sub-Saharan Africa

targets

Name

Name

Name

Name

Name

Name

Name

Name

Name

Name

Name

Name

based-on

Name

StixFile

Value
bc92e8e964e0492b3595d9470e59941bded90082040ac436583b9f3269e1e550

External References

- https://www.zscaler.com/blogs/security-research/dodgebox-deep-dive-updated-arsenal-apt41-part-1
- https://otx.alienvault.com/pulse/668fca957baf422e058e7756