

An Encounter With TA551/Shathak

 blog.reconinfosec.com/an-encounter-with-ta551-shathak

The Recon incident response team recently responded to a case of business email compromise. The incident spanned over seven months of potential dwell time, and included the unraveling of encrypted malware hidden in an image file. Our analysis attributed the incident to a threat group known as TA551/Shathak, known for stealing banking credentials.

Read on to learn more!



FIRST SIGN OF TROUBLE

Most business email compromise investigations seem to start in one of two ways:

1. Someone uncovers a fraudulent wire transfer that leads back to a compromised mailbox
2. A compromised user begins receiving confused responses and bounce-back errors for weird emails they never sent

In this case, we were called in for #2.

To start, all we had were the error and responses back to the compromised user. They told us our victim was sending out messages that looked like this:

Hello ,

The important information for you.

See the attachment to the email.

Password - 1234567

Thanks

Attached: request.zip

We've seen this campaign before; it's TA551/Shathak. We've been quick to catch these malicious emails for existing customers before they can do damage. Unfortunately, this was not an existing customer. Without pre-deployed monitoring and log aggregation tools, we knew we'd have to perform some manual analysis of the affected user's system and email mailbox.

INITIAL ACCESS

We had two plausible hypotheses for initial access. First, as we quickly learned, the victim's password was guessable. Based on the configuration of the email server, anyone could brute force this password from the Internet without any mitigating controls. This would be our best case scenario because it would limit the incident's scope to the compromised email account. Unfortunately, this hypothesis doesn't match the known tactics of TA551/Shathak.

This leads to our second hypothesis: our victim opened a malicious attachment similar to the ones sent from the compromised account. Our victim likely received a similar email some time in the past. A compromised system would increase the scope of our investigation beyond just webmail.

PROVING WORKSTATION COMPROMISE

Our customer asked us to prove malware on the workstation was the initial access point in order to justify the effort and cost of remediating the system. We also wanted to bolster our evidence attributing this incident to TA551/Shathak. At the moment all we really had were the contents of the outbound emails.

Our team uses Velociraptor to interrogate and analyze endpoints. We started by querying for any evidence of `requests.zip` on the user's system. We found it sitting in the downloads folder, created about seven months ago.

This was a quick win, but not a guarantee of a successful compromise. It may be a fair assumption to conclude this downloaded file was our smoking gun, but it's still an assumption. The seven month gap increased the risk that we were looking at a failed first

attempt; the user may not have run this malicious file. We wanted evidence of code execution.

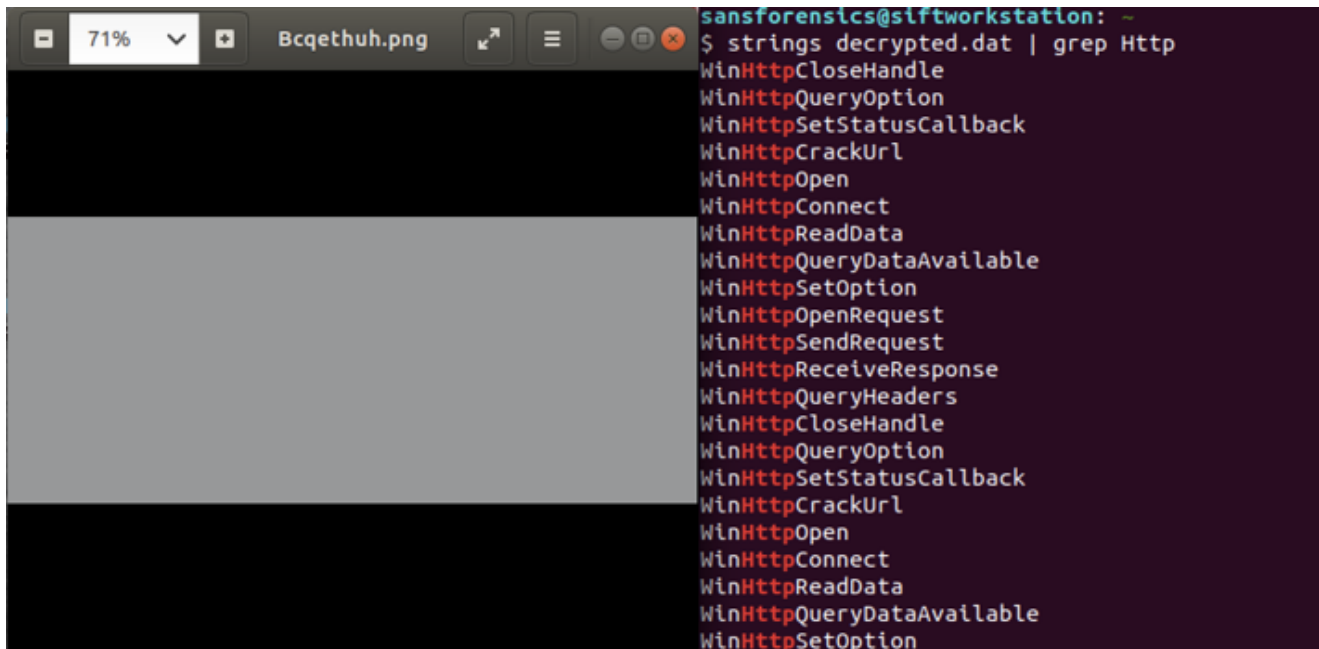
Mimecast's [excellent writeup](#) on the TA551/Shathak and its malware goes into detail about what we should expect to see in the next stage of the attack. The encrypted zip file contains a macro-enabled Office document. According to Mimecast, the malicious document uses a series of C2 channels and droppers to download and execute malware hidden in an encrypted PNG image. This image file, stored somewhere in `C:\Windows\Temp`, became our next target.

DECRYPTING THE MALWARE

Once again, Velociraptor gave us a quick way to search for this PNG file. We found a likely candidate in `C:\Users\{user}\AppData\Local\{GUID}\Bcqethuh.png`. This file was created several months after `requests.zip`, which was unexpected. To eliminate any doubt that this suspicious PNG file was malicious and tied to our TA551/Shathak campaign, we kicked off an effort to decrypt it.

We tinkered with Mimecast's proof-of-concept decryption code, and *poof*, our suspicious PNG revealed itself as executable malware, likely the IceID banking trojan. Our working decryption code is available [on our Github](#).

```
sansforensics@siftworkstation: ~
$ python3 decrypt.py '/home/sansforensics/Desktop/Bcqethuh.png'
Processing png file {filename}
Found IDAT at 87
  Section length: 678565
  Key length: 8
  Hash in image: 3934460640
b'y %\x99\xba"\xc4\x18'
Hash checks out
Results of decrypt.py
```



```
sansforensics@siftworkstation: ~
$ strings decrypted.dat | grep Http
WinHttpCloseHandle
WinHttpQueryOption
WinHttpSetStatusCallback
WinHttpCrackUrl
WinHttpOpen
WinHttpConnect
WinHttpReadData
WinHttpQueryDataAvailable
WinHttpSetOption
WinHttpOpenRequest
WinHttpSendRequest
WinHttpReceiveResponse
WinHttpQueryHeaders
WinHttpCloseHandle
WinHttpQueryOption
WinHttpSetStatusCallback
WinHttpCrackUrl
WinHttpOpen
WinHttpConnect
WinHttpReadData
WinHttpQueryDataAvailable
WinHttpSetOption
```

Encrypted PNG File Reveals Hidden Executable Code

FINDINGS AND NEXT STEPS

With our initial triage complete, we were ready to give our customer an update: this incident actually started several months ago and affected more than just the user's mailbox. The attacker had gained access to the victim user's workstation via a malicious Word document. As a result of this compromise, the IcelD banking trojan was executed on the system resulting in at least the theft of the user's email credentials. These stolen credentials were what the attacker used to send malicious emails to the victim's contacts. Based on our attribution, the attacker was likely looking for banking credentials.

Our tactical recommendations included isolating the affected workstation, resetting the user's credentials (including anywhere this credential was reused), and notifying the employee about the personal risks of the banking trojan so they could take appropriate precautions. We also started the process of scoping the incident, including analyzing the rest of the environment for evidence of the IcelD trojan and other related indicators.

This attack showed the importance of a defense-in-depth strategy. Even best practices against email phishing, like email filtering and attachment analysis, would have failed to prevent this incident. To bypass filtering, the email originated from a compromised but *legitimate* contact. To bypass malware analysis, the attacker encrypted the payload inside a password protected .zip archive. Our last lines of defense, endpoint protection/detection and user awareness training, were in the best position to stop this attack in its tracks. Every layer of defense matters!

ICEID ENDPOINT DETECTION

The following two Sigma rules detect the execution of the IcedID trojan based on its misuse of Regsvr32 to execute malicious code (MITRE ATT&CK T1218.010).

```
title: Suspicious Scheduled Task Creation Leveraging Regsvr32
status: stable
description: Detects the creation of scheduled tasks that leverage regsvr32 to load
malicious dll files
author: Luke Rusten
references:
  - https://www.mimecast.com/globalassets/documents/whitepapers/taa551-
treatresearch_final-1.15.21.pdf
  - https://unit42.paloaltonetworks.com/ta551-shathak-icedid/
tags:
  - attack.persistence
  - attack.t1053.005
  - attack.t1218.010
logsource:
  category: process_creation
  product: windows
detection:
  selection:
    CommandLine|contains|all:
      - 'schtasks'
      - ' /create '
      - 'regsvr32'
  condition: selection
fields:
  - CommandLine
falsepositives:
  - Unknown
level: medium
```

title: Scheduled Task Leveraging Regsvr32
description: Detects scheduled tasks that are leveraging regsvr32 to load malicious dll files
status: stable
author: Luke Rusten
references:
- https://www.mimecast.com/globalassets/documents/whitepapers/taa551-treatresearch_final-1.15.21.pdf
- <https://unit42.paloaltonetworks.com/ta551-shathak-icedid/>
tags:
- attack.persistence
- attack.t1053.005
- attack.t1218.010
logsource:
product: windows
service: security
definition: 'The Advanced Audit Policy setting Object Access > Audit Other Object Access Events has to be configured to allow this detection (not in the baseline recommendations by Microsoft). We also recommend extracting the Command field from the embedded XML in the event data.'
detection:
selection:
EventID: 4698
TaskContent: '*regsvr32*'
condition: selection
falsepositives:
- Unknown
level: medium

LOOKING FOR EXPERTISE?

The Recon team consists of passionate experts that eat, sleep and breathe defensive security operations. If you are looking for a partner, check out our [services](#) or [contact us](#).

Tags: [DFIR](#), [SecOps](#), [Security](#), [TA551](#), [Shathak](#), [Python](#), [Malware](#)



Written by [Andrew Cook](#)

Director of Security Operations