TheHive: Detector de vulnerabilitats en sistemes d'entorns locals

Raquel García Galán

Resum— En els últims anys la major part de la informació que es genera és emmagatzemada en format digital en sistemes que probablement tenen accés a Internet. Qualsevol node que formi part d'aquesta gran xarxa és vulnerable a ser atacat. Amb l'evolució d'Internet i la facilitat d'accés per part de qualsevol usuari produeix que els atacants cada cop estiguin més preparats.

Fa unes dècades, la seguretat informàtica no es considerava com un terme important al que parar atenció. Com és habitual en l'ésser humà, només pren notorietat allò que es pot percebre visualment. No ha estat fins fa uns anys, i ara amb l'accentuació de l'ús massiu de les xarxes socials on els usuaris han compartit informació indiscriminadament a través de la xarxa, que no s'ha pres consciència del perill real que corren les dades al viatjar a través de la xarxa. És per això que la seguretat informàtica ha esdevingut en els últims anys un àmbit crucial en el qual estan invertint empreses, institucions i organitzacions. Ara les organitzacions són conscients de que la informació és un dels béns amb més valor que posseeixen i que ha de ser protegida.

Paraules clau— Seguretat, vulnerabilitats, xarxa, host, port, sistema operatiu, servei, fingerprinting, TCP/IP, sockets, XML, SQLite3, Python.

Abstract— In recent years most of the information generated is stored in digital systems that probably have Internet access. Any computer that is part of this great network is vulnerable to be attacked. The Internet evolution and the ease of access by any user, allow the attackers to be more prepared .

A few decades ago people didn't see security as an important term to pay attention. As is common, humans pay attention only to what can be seen. It was not until a few years ago, and now with the massive use of social networks where users share a lot of information through the net, that people haven't realized the real danger of data travelling through the vast network. That's why computer security has become in recent years a crucial area in which enterprises, institutions and organizations are investing. They are aware that the information is most valuable properties that possess and must be protected.

Index Terms— Security, vulnerabilities, network, host, port, operation system, service, fingerprinting, TCP/IP, sockets, XML, SQLite3, Python.

1 Introducció

La seguretat és un tema que s'ha posat clarament sobre la taula en els últims anys. Cada cop hi ha més demanda de personal qualificat en aquesta àrea de coneixement per part d'empreses i organitzacions.

Les entitats que cal protegir no són d'una única naturalesa. Aplicacions web, servidors, bases de dades o ordinadors d'usuaris amb informació confidencial, entre d'altres, poden ser els objectius dels atacants. Tota aquesta aglomeració de possibilitats es pot classificar en tres àmbits d'actuació [1]:

- **Extern**. Engloba qualsevol xarxa externa a la nostra.
- **Intern**. Equival a l'àmbit de la nostra xarxa.
- **Local**. Fa referència a un sistema que forma part de la xarxa interna.

Cal destacar que la major part de la seguretat recau sobre l'àmbit extern. La gran preocupació acostuma a ser evitar que un agent extern aconsegueixi accedir a la xarxa interna. Això molt sovint provoca que es descuidin els altres dos àmbits. És llavors quan apareixen els problemes. Si un atacant aconsegueix saltar la primera barrera ja pot accedir a qualsevol recurs perquè els altres dos àmbits solen estar més desprotegits. S'ha de prendre consciència d'això.

La seguretat d'una xarxa la podem definir, a grans trets, com la que s'encarrega de garantir els aspectes de la informació descrits a continuació [2]:

- *E-mail de contacte: raquel.garcia@e-campus.uab.cat*
- *Menció realitzada: Tecnologies de la Informació.*
- *Treball tutoritzat per: Àngela Fabregues Vinent (Departament d'Enginyeria de la Informacio i de les Comunicacions)*
- *Curs 2015/16*

- **Integritat**. Assegurar que les dades no es puguin modificar per tercers no autoritzats.
- **Confidencialitat.** Assegurar que els usuaris puguin accedir als recursos que corresponguin al seu nivell de permisos.
- **No-repudi**. Assegurar que un usuari que interactua amb la xarxa no pugui desmentir una acció feta.
- **Accessibilitat.** Assegurar que els recursos puguin estar disponibles per a l'accés de l'usuari.
- **Autenticació**. Assegurar que l'usuari és qui diu ser.

No hi ha una única manera de salvaguardar aquests aspectes. És més, realment, els atacants no segueixen cap manual d'actuació sinó que busquen falles existents i les aprofiten per penetrar dins dels sistemes que formen part de la xarxa.

Aquest projecte es centra en la cerca i detecció de vulnerabilitats existents en l'àmbit intern, per tal de corregir-les i evitar així que puguin ser explotades.

Aquest article està dividit en diferents apartats. En primer lloc es presenten els objectius, on es concreten els objectius proposats de manera seqüencial. Seguidament s'explica l'estat de l'art, on es descriu el context del projecte i els conceptes relacionats. A continuació s'exposa la metodologia utilitzada amb la qual s'ha desenvolupat el projecte. En els apartats TheHive, NetworkMapper, PortScanner, ProcManager, VulnDetector i Implementació, s'explica el resultat de la realització d'aquest projecte en relació als objectius proposats. Després es detallen alguns aspectes puntuals del projecte com la gestió de processos, la paral·lelització i l'optimització a l'accés a base de dades. Seguidament s'exposen els experiments realitzats i els seus resultats. I per acabar, a l'apartat de conclusions es valora l'assoliment dels objectius i s'expliquen les possibles extensions del projecte.

2 OBJECTIUS

El principal objectiu d'aquest projecte és detectar vulnerabilitats a les diferents màquines que formen part d'una xarxa d'àrea local.

D'aquest objectiu es poden desprendre un conjunt de subobjectius que corresponen a cadascuna de les funcionalitats que duu a terme l'aplicació. Són els següents:

- **Mapeig de la xarxa.** Identificació de les màquines actives de la xarxa.
- **Mapeig de ports.** Descobriment de l'estat dels ports de cada màquina activa trobada anteriorment.
- **Recol·lecció d'informació de la xarxa.** Obtenció d'infomació relacionada amb el sistema operatiu i les versions dels serveis que corren a cada màquina.
- **Detecció de vulnerabilitats.** Identificació de les vulnerabilitats presents a cadascuna de les màquines actives en relació a la informació trobada.

Com tot treball final de grau, amb aquest projecte es pretén assentar i aplicar els coneixements adquirits al llarg de la menció, en aquest cas relacionada amb l'àrea de xarxes i seguretat.

3 ESTAT DE L'ART

Actualment existeixen una gran quantitat d'eines en aquesta àrea de coneixement. Les eines existents, segons *the project NMAP* [3], es poden classificar en:

- **Antimalware.** Anàlisis en busca de *malware*, software maliciós, al sistema i esborrar-los sota ordres de l'usuari.
- **Application-specific scanners.** Detecció de les aplicacions que s'estan executant en un host i en quin port ho estan fent.
- **Web browser–related.** Eines relacionades amb els navegadors web.
- **Encryption tools.** Encriptació i desencriptació informació emmagatzemada o comunicada.
- **Debuggers.** Depuració d'aplicacions ja compilades per tal de fer enginyeria inversa, obtenir codi font a partir de compilats.
- **Firewalls.** Filtració el tràfic segons regles de comunicació definides.
- **Forensics.** Obtenció, identificació i anàlisi de dades residuals o equips informàtics (informàtica forense).
- **Fuzzers.** Detecció de vulnerabilitats en aplicacions a base d'introduir dades d'entrada i buscar possibles errors o excepcions.
- **Intrusion detection Systems.** Identificació d'intrusions a la xarxa en temps real.
- **Packet crafting tools.** Construcció i enviament de paquets TCP o UDP a través de la xarxa.
- **Password auditing.** Avaluació de la fortalesa de les contrasenyes d'usuari en organitzacions.
- **Port scanners.** Detecció dels estats dels ports d'una màquina.
- **Rootkit detectors.** Detecció i eliminació de rootkits, programes que tenen com a objectiu permetre un accés de privilegi continu en un sistema.
- **Security-oriented operating Systems.** Sistemes operatius destinats a ser utilitzats en l'àmbit de la seguretat ja que, entre d'altres coses, contenen per defecte una biblioteca d'eines de tots els tipus.
- **Packet Sniffers.** Captura del tràfic de la xarxa i anàlisi de les trames.
- **Vulnerability exploitation tools.** Anàlisi i explotació de vulnerabilitats, a l'àmbit del *pentesting*.

- **Traffic monitoring tools.** Monitorització del tràfic de la xarxa. Utilitzat per administradors de sistemes.
- **Vulnerability scanners.** Detecció de vulnerabilitats a nivell d'aplicació, de xarxa o aplicació.
- **Web proxies.**
- **Web vulnerability scanners.** Identificació de possibles vulnerabilitats a l'arquitectura d''aplicacions web a través del frontend.
- **Wireless tools.** Eines relacionades amb xarxes sense fils que estan, basades en l'estàndard 802.11.

Per preservar la seguretat i desenvolupar una bona tasca és clau identificar el problema correctament per saber quina eina s'ha d'utilitzar. El més important és entendre el que s'està fent i el que es vol fer.

L'eina desenvolupada en aquest projecte és clarament del tipus escàner de vulnerabilitats, encara que s'ha fet ús d'utilitats d'altres tipus com és el cas d'*NMAP* [4] o *Wireshark* [5].

4 METODOLOGIA

La metodologia utilitzada en el projecte és Scrum[6][7][8]. És també una de les metodologies denominades àgils, principals característiques de les quals són les següents:

- L'estratègia de desenvolupament és incremental, és a dir, les tasques es desenvolupen segons les necessitats del client durant cada Sprint (període de temps definit amb el client, en el nostre cas: 2 setmanes). D'aquesta manera aconseguim un feedback del client que beneficia i optimitza el desenvolupament del producte que es podria abastar si es fa una planificació i execució completa del producte.

- La qualitat del resultat va lligada al coneixement implícit de les persones en equips auto organitzats en comptes d'en la qualitat dels processos que es duen a terme.

- Les diferents fases del desenvolupament se superposen. Això permet una planificació flexible en què els possibles canvis en les prioritats del client permeten avançar tasques assignades a un sprint posterior o ajornar tasques, que han deixat de ser urgents, a sprints posteriors. Realitzant una tasca darrere d'una altra en un cicle seqüencial o en cascada no és possible.

Escollim aquesta metodologia perquè aporta una sèrie d'avantatges que enumerem a continuació:

- Entrega de resultats cada quinze dies, que corresponen als requisits completats més prioritaris en aquell moment.
- Gestió regular de les expectatives del client i basada en resultats tangibles. Cada entrega aporta millores al projecte.
- Resultats anticipats, el que anomenem time to market.
- Flexibilitat i adaptació respecte a les necessitats del client, canvis en el mercat, etc.
- Mitigació sistemàtica dels riscos del projecte.
- Alta productivitat i qualitat.
- Alineament entre el client i l'equip de desenvolupament que facilita les comunicacions.

Al llarg del projecte s'han anat realitzant diversos sprints, tots ells documentats i inclosos al document Informe inicial [9] i en detall al document Sprints[10].

5 THEHIVE

TheHive és una eina de detecció de vulnerabilitats que treballa en l'àmbit local, és a dir, en executar-se detecta les vulnerabilitats de les màquines que formen part d'una xarxa local. En cada execució es realitzen els següents processos:

- **Mapeig de la xarxa.** S'identifiquen les màquines actives de la xarxa i es mostren les seves adreces per pantalla.

- **Mapeig de ports.** Es descobreixen els ports, i els seus estats, de cadascuna de les màquines actives. Per pantalla es mostra l'adreça de la màquina, el número de cada port identificat, acompanyat del nom del servei, la descripció del servei i l'estat del port.

- **Recoleccio d'informacio de la xarxa.** S'extreu informació relacionada amb els serveis dels ports i el sistema operatiu de cada máquina, a partir de l'execució d'un conjunt de funcions. Per a cada una d'elles, es mostra per pantalla el titol i la informació trobada.

- **Deteccio de vulnerabilitats.** S'identifiquen i es mostren per pantalla les vulnerabilitats presents a cada màquina a partir de la informació trobada al procés anterior.

TheHive és una eina desenvolupada modularment, per tal de posibilitar la seva exrensibilitat, tot definint un mòdul per procés. Això fa que en un futur pugui créixer i pugui adoptar noves funcionalitats sense necessitat de modificar l'aplicació.

6 NETWORKMAPPER

El NetworkMapper és el mòdul que s'encarrega de fer un mapping de la xarxa a la que es troba connectada la màquina on s'executa l'aplicació.

En primer lloc llegeix la taula d'encaminament i extreu la direcció IP i la màscara del router. A partir d'aquesta direcció es calcula la direcció de la xarxa.

A continuació es fa PING, s'envia un missatge Echo Request, a cadascuna de les direccions que formen part del rang de direccions de la xarxa.

Finalment, es comprova la resposta i en cas de que sigui un missatge ICMP diferent al tipus 3 (Destination unreachable), vol dir que la màquina està activa i per tant es guarda la seva adreça..

7 PortScanner

El PortScanner és el mòdul que s'encarrega de mapejar els ports de totes les màquines actives de la xarxa. Aquest rep una direcció IP corresponent a una màquina activa i s'encarrega de mapejar els 1000 primers ports d'aquesta. Es va decidir aquest número inicialment ja que conté els ports dels serveis més importants exceptuant algun cas i així s'ha implementat.

Mitjançant sockets s'estableix una connexió amb la màquina per a cada un dels ports. S'analitzen els paquets de resposta i s'identifica l'estat del port corresponent. Aquest anàlisi es realitza a través de tècniques basades en enviar paquets i analitzar la resposta [11][12]. Són les següents:

- **TCP Connect Scan.** Mètode que comprova l'estat del port a través del procés de connexió 3-way-handshake. S'envia un paquet amb el flag SYN activat, el host respon amb SYN+ACK i retorna ACK. Depenent de si rep resposta o no, i de quins flags té activats estarà obert o tancat.

- **TCP Stealth Scan**. És igual que el mètode anterior però amb la diferencia que no completa la connexió. Només avalua la primera resposta. Amb aquests ja podem saber si està obert, tancat o filtrat.

- **ACK Scan**. S'envia un paquet amb el flag ACK activat. Depenent de si rep com a resposta RST, de si no en rep o si és ICMP, voldrà dir si està o no filtrat.

- **FIN Scan**. S'envia un paquet amb el flag FIN activat. Si no hi ha resposta el port estarà obert, i si està tancat es rebrà un paquet amb el flag RST.

- **XMAS Scan.** S'envia un paquet amb els flags PSH, FIN i URG. Si està obert no hi haurà resposta, si està tancat es rebrà un paquet amb el flag RST i si està filtrat un paquet ICMP de tipus 3.

- **NULL Scan**. S'envia un paquet sense cap flag activat. Si el port està obert el host remot no enviarà resposta. Si està tancat es rebrà un paquet amb RST activat.

8 ProcManager

El ProcManager és el mòdul que s'encarrega de recol·lectar informació de la xarxa. Aquesta informació fa referència a dos atributs: el sistema operatiu i els serveis dels ports. S'han definit dos mètodes per a cadascun d'aquests dos atributs. Els mètodes que tenen com a objectiu trobar el sistema operatiu són :

- **OS fingerprinting.** S'analitzen els paquets que provenen del host target i els camps del paquet TCP. Hi ha camps de la capçalera TCP o IP que venen definits per defecte depenent del sistema operatiu per tant amb l'anàlisi d'alguns d'aquests paquets podem intuir quin sistema operatiu està corrent.

- **Test del Port 0.** Com bé és sabut si intentem enviar un paquet amb el port emissor com a 0 o s'intenta assignar 0 a un socket en escolta, el sistema re-assigna aquest valor agafant qualsevol dels ports lliures. Això passa perquè és un port reservat i té un propòsit concret tal i com s'explica el RFC 1700 [16].

  Aquest mètode es pot definir com un cas concret de OS fingerprinting. Consisteix en un test de 7 passos [17] on cada pas equival a un enviament d'un paquet de caire diferent.

  Depenent del sistema operatiu respon d'una manera determinada a cada enviament de cada pas d'aquest test.

```
P1: send tcp packet from source port 0 to port 0
P2: send tcp packet from source port X to port 0
P3: send tcp packet from source port 0 to open port
P4: send tcp packet from source port 0 to closed port
P5: send udp packet from source port 0 to port 0
P6: send udp packet from source port 53 to port 0
P7: send udp packet from source port 0 to closed port

Port X in test P2 is any port not equal to 0. Port 53
```

Fig 1: 7 passos del Test del Port0

Els mètodes que tenen com a objectiu trobar la versió dels serveis són:

- **Banner Grabbing.** Esbrina la informació d'un servei a partir de la capçalera [12][18]. S'aplica una variació d'aquest mètode.
  A partir del nom dels serveis carregats per defecte, definits per *l'Internet Assigned Numbers Authority* (IANA) [19], amb aquest mètode es comproba si realment està corrent el servei assignat per defecte al port per defecte.

- **NMAP version detection.** Per cada port sense

versió de servei informat s'executa una comanda NMAP per obtenir aquesta informació [20].

El ProcManager s'encarrega de la gestió de l'execució dels mètodes anteriors. Decideix quin mètode s'ha d'executar en base a la informació que es va obtenint de la xarxa. Això és necessari ja que per exemple si després d'executar l'OS fingerprinting s'han informat els sistemes operatius de les màquines és redundant i innecessari executar el Test del Port0, que té la mateixa finalitat, en aquest cas.

En un futur el número de mètodes a executar podria augmentar o podrien aparèixer nous atributs a informar com, per exemple, el tipus de dispositiu del que es tracta. És per això que és de vital importància que el ProcManager tingui aquesta capacitat de decisió i només s'executin aquells mètodes que siguin realment necessaris per aportar la informació que no es té.

9 VULNDETECTOR

El VulnDetector és el mòdul que s'encarrega de la detecció de vulnerabilitats de cada màquina activa. A partir de la informació aconseguida anteriorment cerca vulnerabilitats a la base de dades de l'aplicació, que està basada en la *National Vulnerability Database (NVD)* [21].

10 IMPLEMENTACIÓ

.L'eina theHive s'ha implementat en un entorn Linux amb ditribució Ubuntu 14.04 amb Python 2.7, fent servir un conjunt de llibreries, de les quals les més importants són: *xml.etree.ElementTree*[22], *sqlite3*[23], *multiprocessing* [24], *subprocess*[25], netaddr[26], scapy[27], socket[28], threading[29], queue[30] i os[31].

L'estructura de la informació amb la que treballa l'aplicació s'ha representat en el següent model de dades:

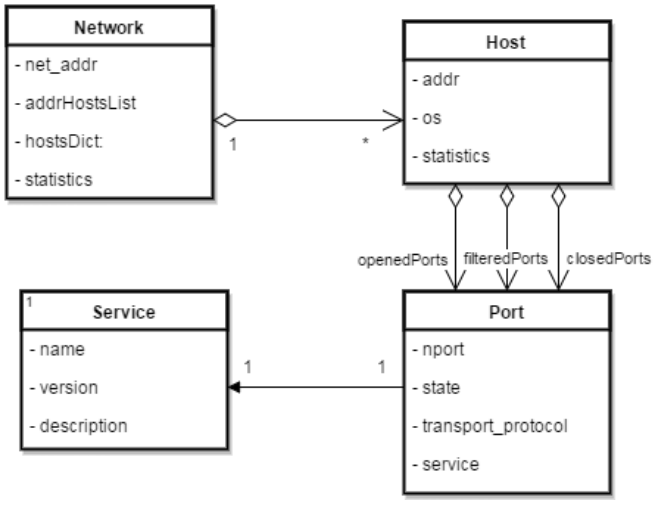

Fig. 2: Diagrama de classes de l'aplicació

- **Network.** Representa la xarxa sobre la que estem treballant. Els aspectes més destacables d'aquesta classe són els atributs *hostsDict*, un diccionari on accedim als objectes de tipus Host a partir de la seva adreça IP coma string (cada objecte Host representa una màquina activa de la xarxa); i la propietat *statistics* on es guarden les estadístiques sobre el sistema operatiu de les màquines que componen la xarxa.

  Aquesta estadística guarda una llista amb 3 valors: el nombre total de màquines actives, el nombre de màquines les quals es coneix el sistema operatiu i una llista amb els objectes *Host* corresponents a aquelles màquines de les que no es coneix el sistema operatiu.

  Aquestes estadístiques són necessàries per mantenir un control de l'èxit de l'aplicació però també un control necessari perquè el mòdul que treballa amb aquesta informació, el *ProcManager*, sàpiga a cada moment l'estat d'aquesta informació.

- **Host.** Representa cada màquina que pertany a la xarxa. D'aquesta classe cal comentar que es guarden els ports en tres llistes diferents depenent del seu estat. S'ha pres aquesta decisió ja que la majoria dels casos en que es tracta amb objectes Port es treballa diferent segons el seu estat. Això també accelera l'obtenció d'un port si el volem cercar.

  En aquesta classe també guardem estadístiques, en aquest cas sobre la informació dels serveis.
  És una llista que conté el nombre de ports oberts totals, el nombre de ports dels que es coneix el nom del servei i la versió, i una llista dels objectes Port que corresponen als ports dels qual no es té la informació completa.

- **Port.** Aquesta classe representa els ports de les màquines.

- **Service.** Aquesta classe representa el servei que corre darrere d'un port.

11 GESTIÓ AUTOMATITZADA DE PROCESSOS

Anteriorment s'ha descrit la tasca que duu a terme el ProcManager. A continuació s'explica com s'ha implementat aquesta capacitat de gestió automatitzada.

Per a cada execució, el ProcManager es construeix una cadena boleana. Cada posició representa un atribut, en el nostre cas en tenim 2: el sistema operatiu, i la versió dels serveis. El mòdul és coneixedor de la posició a la que correspon cada atribut. El valor de cada posició de la

cadena serà *True* o *False* depenent de si cal trobar la informació de l'atribut o d'altra banda ja la tenim, respectivament. En el nostre cas com només tenim dos atributs, la cadena només té dos posicions.

Inicialment, i a partir de les estadístiques de l'objecte Host, el ProcManager construeix aquesta cadena assignant *True* o *False* a cada posició segons escaigui.

A continuació evalua aquesta cadena i extreu les posicions que són *True*, que representen els atributs dels quals no tenim informació o no està completa. Com el mòdul coneix la posició que ocupa cada atribut, a partir de les posicions obtingudes el ProcManager es guarda el nom dels atributs que cal informar.

Seguidament el ProcManager executa un mètode master on de manera automàtica es realitzen els mètodes de cada atribut guardat anteriorment. Dins de l'aplicació, a cada atribut li correspon un directori, on es guarden els mètodes que s'han d'executar. Cada mètode està definit a un fitxer diferent. Tots els fitxers tenen en comú que contenen un mètode *run*(), aquest mètode s'encarrega de cridar tota la funcionalitat. Per tant el que fa el mètode master del ProcManager és a partir del nom de l'atribut, es posiciona en el directori corresponent i llegeix i emmagatzema el nom dels fitxers que conté. Finalment importa i executa el mètode *run()* de cadascun dels fitxers.

Al final de l'execució cada mètode es modifiquen les estadístiques en funció de si s'ha aconseguit trobar aquesta informació o no, i es torna a construir la cadena boleana. Això es fa per evitar l'execució de tots els mètodes d'un atribut si la informació ja s'ha trobat.

S'ha dissenyat d'aquesta manera per poder aconseguir que l'aparició de nous atributs o nous mètodes no impliqui greus canvis a la implementació, cosa que permet un fàcil manteniment.

12 Optimització de l'accés a NVD

Per tal d'emmagatzemar la gran quantitat d'informació que representen les vulnerabilitats s'ha trobat oportú crear una base de dades. Aquesta informació inicialment està continguda en un arxiu XML facilitat pel repositori NVD. Accedir a llegir a aquest arxiu és molt costós ja que pesa molt, d'aquí la nostra decisió.

En aquesta base de dades podem obtenir l'identificador de la vulnerabilitat (CVE)[33], una descripció dels detalls de la vulnerabilitat, una llista amb el nom dels products, és a dir, serveis, frameworks o sistemes afectats; i una url pertanyent a la web *https://**nvd**.nist.gov* on es mostren els seus detalls.

13 Paral·lelització

El problema més gran detectat en aquest tipus d'aplicacions és la quantitat de temps que es triga en executar els processos. En situacions com la de fer mapping de hosts i, sobretot, de ports s'ha de tenir en compte que els bucles poden ser de milers d'iteracions i, a més, per cada iteració s'han de fer operacions amb la xarxa cosa que implica que la resposta no és immediata i de vegades ni tan sols hi ha resposta. És per això que s'ha hagut de fer ús de la paral.lelització de processos[32] i s'han utilitzat threads, combinats amb cues.

En molts dels processos que duu a terme l'aplicació s'ha de guardar informació i en tots els casos això es realitza a través de funcions on llegim, modifiquem i guardem. Aquests tres processos individualment són atòmics però, com és evident, no ho són per separat. Nosaltres necessitem que les funcions que contenen aquests tres processos actuïn com si fossin atòmiques ja que la concurrència pot provocar problemes d'integritat de dades. És per això que en aquests casos hem hagut d'implementar semàfors, cosa que en algunes situacions retrassa el procés.

A l'hora de definir el nombre de threads a utilitzar s'han fet proves per tal de verificar que l'execució de l'aplicació no ofega el processador. És important, per tant, prendre nota de l'entorn en què s'ha desenvolupat. En el nostre cas: 4 nuclis d'execució de 2.2GHz.

14 Experiments

Per a comprovar el funcionament de l'aplicació en primer lloc s'ha executat un conjunt de tests funcionals on es verifica el correcte funcionament de cadascun dels mòduls. En segon lloc s'han fet diferents proves de l'eina en diferents entorns.

14.1 Tests funcionals

Per a verificar que un mòdul funciona correctament el que s'ha fet ha estat comprovar cadascun dels mètodes que el conformen.

Tots els tests han donat un resultat satisfactori, aquest s'ha recollit al document Tests [34].

14.2 Experiment I

Aquest experiment s'ha dut a terme a l'empresa K-LAGAN, a la xarxa d'un dels departaments.

L'entorn on s'ha executat és un Linux Ubuntu 15. El resultat de l'execució ha estat el següent:

Nº màquines trobades / Nº màquines actives	4 / 27
Nº de vulnerabilitats trobades	0
Temps total trigat	19 m 07 s

14.3 EXPERIMENT II

Aquest experiment s'ha dut a terme a una xarxa domèstica.

L'entorn on s'ha executat és un Linux Ubuntu 14.04. El resultat de l'execució ha estat el següent:

Nº màquines trobades / Nº màquines actives	9 / 9
Nº de vulnerabilitats trobades	5
Temps total trigat	25 m 34 s

14.3 EXPERIMENT III

Aquest experiment s'ha dut a terme a una xarxa domèstica.

L'entorn on s'ha executat és un Linux Ubuntu 14.04. El resultat de l'execució ha estat el següent:

Nº màquines trobades / Nº màquines actives	5 / 5
Nº de vulnerabilitats trobades	3
Temps total trigat	26 m 49 s

14.3 EXPERIMENT IV

Aquest experiment s'ha dut a terme a una xarxa domèstica.

L'entorn on s'ha executat és un Linux Ubuntu 14.04. El resultat de l'execució ha estat el següent:

Nº màquines trobades / Nº màquines actives	6 / 6
Nº de vulnerabilitats trobades	4
Temps total trigat	14 m 54 s

15 RESULTATS DELS EXPERIMENTS

L'èxit d'aquesta eina resideix en gran part en la informació de la que es disposa. En aquest projecte s'han implementat un numero simbòlic de tècniques. Ens hem centrat més en dissenyar un model base que permetés mostrar el funcionament de l'eina. Per tant els resultats obtinguts s'han de valorar proporcionalment a la informació de la que som capaços d'obtenir actualment.

Tenint en compte aquest punt podem dir que certament aquesta eina és capaç de trobar qualsevol vulnerabilitat que estigui basada en un sistema operatiu concret o un servei determinat, que són els atributs que contemplem.

Des del principi es va ser conscient d'aquest fet per tant podem dir que els resultats que esperàvem han estat els resultats obtinguts.

Els quatre experiments han estat satisfactoris encara que cal fer algunes reflexions en algun d'ells, com és el cas de l'experiment I. En aquest test l'eina no ha trobat totes les màquines possibles. Això és degut a la configuració de la xarxa. La màquina on s'ha executat l'aplicació és una màquina d'usuari, i aquestes no poden establir connexió amb la resta de màquines d'usuari de la planta. Aquest resultat és positiu ja que vol dir que realment la seguretat és un tema que preocupa dins de l'empresa i s'han pres mesures.

En tots els casos el nombre de vulnerabilitats trobades és el màxim nombre de vulnerabilitats que podem trobar per tant podem categoritzar d'èxit aquests experiments.

Finalment cal mencionar que els temps obtinguts en cada cas són variables ja que depenen del nivell de càrrega de la xarxa.

16 CONCLUSIONS

Pel que fa als objectius, per una banda, en aquest projecte s'han pogut plasmar coneixements adquirits i ha permès l'aprenentatge de nous conceptes.

D'altra banda, durant la realització del projecte s'ha profunditzat en els coneixements sobre l'arquitectura de xarxes en entorns empresarials.

Aquest projecte pretenia ser part d'un procés de la securització de la xarxa de l'empresa on estem treballant. Aquest objectiu no s'ha pogut assolir ja que aquest procés no s'ha dut a terme finalment.

El resultat del projecte ha estat satisfactori en la seva majoria. S'han implementat totes les funcionalitats previstes a excepció de la interfície gràfica, que en última instància s'havia definit com no necessària. Aquesta situació s'ha produït degut a un solapament bastant important de feina en aquests últims mesos aliena a aquest projecte.

Per acabar m'agradaria comentar que la realització d'aquest projecte ens ha permès viure en primera persona com és la realització d'un projecte d'enginyeria pas per pas, cosa que ens ajuda de cara a enfrontar-nos-hi al món laboral.

17 LÍNIES FUTURES DE DESENVOLUPAMENT

Aquest projecte pot créixer en diversos aspectes ja que s'ha desenvolupat una base molt sòlida en la que es poden afegir tant nous mòduls com noves funcionalitats sense la necessitat de fer canvis a nivell arquitectural o funcional.

La tecnologia evoluciona a marxes forçades i és per això que ha de ser flexible i permetre incloure noves implementacions. En primer lloc, seria una molt bona opció la implementació de noves tècniques que fessin possible la

identificació de més informació de la xarxa. Com més voluminosa sigui aquesta més acurada serà la cerca de vulnerabilitats.

D'altra banda, aquesta eina esta implementada per a poder mapejar només una xarxa i seria interessant poder analitzar les diferents xarxes a les quals pertany la màquina on s'executa l'aplicació.

Aquesta eina està implementada per a ser executada a nivell de consola. Seria interessant també desenvolupar una interfície gràfica que facilités a l'usuari l'execució i la visualització dels resultats.

AGRAÏMENTS

En primer lloc, m'agradaria agrair el suport rebut a la meva família, especialment als meus pares. Sempre m'han recolzat en les meves decisions i han fet i donat tot per donar-me un futur. En segon lloc a la meva parella que m'ha ajudat i escoltat en tot moment i m'ha fet molt més planer aquest camí, que ha estat difícil. En tercer lloc, a les meves amigues, amb qui m'he hagut de perdre bastantes coses degut a la quantitat de feina.

També m'agradaria agrair a K-LAGAN haver-me donat la opció de desenvolupar aquest projecte tan interessant; als companys de feina Jesús, Gabri i Jordi que m'han ajudat a acabar de perfilar la idea resolent dubtes que m'han aparegut durant l'elaboració del projecte; i a l'Àngela, la meva tutora, pel guiatge documental durant aquests mesos.

REFERÈNCIES

[1] Slides GIS-02: Vulnerability management. Garantia de la Informaci_o i Seguretat[102757]2015

[2] Slides GIS-01: Basic definitions. Garantia de la Informacio i Seguretat [102757] Departament d'Enginyeria de la Informació i les Comunicacions. Universitat Autonoma de Barcelona.2015.

[3] NMAP Porject. SecTools.Org: Top 125 Network Security Tools. [Consultat: 06 de Juny del 2016] Disponible a: http://sectools.org/

[4] https://nmap.org/book/man.html#man-description

[5] https://www.wireshark.org/docs/wsug_html_chunked/ChapterIntroduction.html#ChIntroWhatIs

[6] Schwaber, K y Sutherland, J. (2013), La Guía de Scrum. Scrum.Org and ScrumInc.

[7] Palacio, J., (2014), Gestiónde proyectos Scrum Manager, Scrum Manager. Referència 2

[8] Beneficios de Scrum. (2008), [online] Proyectos Ágiles. Disponible a:http://proyectosagiles.org/beneficios-de- scrum/ [Últim accés: 1 Mar. 2016].

[9] Garcia, R., (2016). Informe de progrés I. ETSE UAB.

[10] Garcia, R., (2016). Sprints. ETSE UAB.

[11] O'Connor, TJ., (2013). Violent Python A Cookbook for Hackers, Forensic Analysts, Penetration Testers and Security Engineers. Elsevier.

[12] Echeverri, D., (2014). Python para Pentesters. ZeroXWord.

[13] EC-Council, (2011). Penetration Testing, Procedures & Methodologies. EC-Council.

[14] Raj, M., (2015). Python Penetration Testing Essentials. Packt Publishing Ltd.

[15] Young, S., Aitel, D., (2004). The Hacker's Handbook, The Strategy Behind Breaking Into And Defending Networks. CRC Press LLC.

[16] K. Reynolds, Joyce., RFC 1700. ASSIGNED NUMBERS. October 1994. [Consultat: 13 de abril del 2016] Disponible a: https://www.ietf.org/rfc/rfc1700.txt

[17] Ste Jones. Network Penetration. [Consultat: 14 de Maig de 2016] Disponible a: https://www.gont.com.ar/docs/port-0-os-fingerprinting.txt

[18] Gregg, M,. Watkins, S., Mays, G., Ries, C., Bandes, R., Franklin, B., (2006). Hack the Stack. O'Reilly Media.

[19] IANA. Service Name and Transport Protocol Port Number Registry. [Consultat el: 16 d'Abril de 2016] Disponible a: https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xml

[20] Chapter 7. Service and Application Version Detection. Nmap Network Scanning. [Consultat el: 24 d'Abril de 2016] Disponible a: https://nmap.org/book/vscan.html

[21] National Vulnerability Database. . [Consultat el: 30 d'Abril de 2016] Disponible a: https://nvd.nist.gov/

[22] 19.7.xml.etree.ElementTree — The ElementTree XML API. [Consultat el: 14 de març de 2016] Disponible a: https://docs.python.org/2/library/xml.etree.elementtree.html

[23] 11.13. sqlite3 — DB-API 2.0 interface for SQLite databases. [Consultat el: 16 de maig de 2016] Disponible a: https://docs.python.org/2/library/sqlite3.html

[24] 16.6. multiprocessing — Process-based "threading" interface. [Consultat el: 07 de març de 2016] Disponible a: https://docs.python.org/2/library/multiprocessing.html

[25] 17.1. subprocess — Subprocess management. [Consultat el: 03 de març de 2016] Disponible a: https://docs.python.org/2/library/subprocess.html

[26] netaddr 0.7.18. [Consultat el: 04 de març de 2016] Disponible a: https://pypi.python.org/pypi/netaddr

[27] Starting Scapy. [Consultat el: 18 de març de 2016] Disponible a: http://www.secdev.org/projects/scapy/doc/usage.html

[28] 17.2. socket — Low-level networking interface. [Consultat el: 08 de maig de 2016] Disponible a: https://docs.python.org/2/library/socket.html

[29] 16.2. threading — Higher-level threading interface. [Consultat el: 17 de maig de 2016] Disponible a: https://docs.python.org/2/library/threading.html

[30] 8.10. Queue — A synchronized queue class. [Consultat el: 17 de maig de 2016] Disponible a: https://docs.python.org/2/library/queue.html

[31] 15.1. os — Miscellaneous operating system interfaces. [Consultat el: 21 de maig de 2016] Disponible a: https://docs.python.org/2/library/os.html

[32] Venthur, B., Writing Concurrent Applications in Python. Institute of Technology. Berlin. 2011-09-14.

[33] About CVE Identifiers. [Consultat el: 21 de maig de 2016] Disponible a: https://cve.mitre.org/cve/identifiers/

[34] Garcia, R., (2016). Tests funcionals. ETSE UAB.