

Weboldalak intelligens szimulálása

DUNAÚJVÁROSI FŐISKOLA

2011. július 18.
Szerző: Horváth Botond

Weboldalak intelligens szimulálása

Tartalomjegyzék

Tartalomjegyzék	1.
1. Kivonat	2.
2. Bevezetés	3.
3. A rendszer működési elve és telepítésének lépései	4.
3.1 A proxy szerver tisztatelepítése	6.
3.1.1 A hálózati konfigurálása	9.
3.1.2 A rendszer kezdeti paramétereinek beállítása	14.
3.1.3. A jelenlegi rendszerhez tartozó fájlok és könyvtárak listája	14.
3.1.4. A system.pl szkript paraméterei	16.
4. A kliens tisztatelepítése és konfigurálás	18.
4.1 A rendszer használata és konfigurálása	20.
5. A vizsgálati környezet strukturális felépítése és tesztelési módszere, valamint felhasználása fontossága	23.
5.1 A vizsgálati környezet automatikus működésének folyamatábrája	27.
6. A rendszer fejleszthetősége és jelenlegi fejlesztései	30.
7. Összefoglalás	31.
8. Felhasznált irodalom	32.

Weboldalak intelligens szimulálása

1. Kivonat

Számítógép, PDA, telefon. A közös ezekben az eszközökben a böngésző, illetve az internet-csatlakozás megléte, amivel a világhálóra képesek csatlakozni. Ez a technikai fejlődés újfajta veszélyeket is hordoz. Manapság a kártevők nem adathordozókon vagy fertőzött programokon keresztül terjednek, hanem kártékony weboldalak, kártékony hivatkozások formájában. A drive-by-download alapú támadások száma az elmúlt időszakban folyamatosan emelkedik és a kutatókat újfajta megközelítésre sarkalja. A legelterjedtebb kártevők megjelenésének pusztá követése, illetve archiválása nem egyszerű feladat. Kutatásom során egy olyan intelligens rendszert valósítottam meg, mellyel lehetséges az interneten tárolt kártékony oldalak felderítése és archiválása. A dolgozatomban kitérek a teljes rendszer, illetve az egyes részegységek működésére, a fejlesztés korai stádiumában fellépő nehézségekre és a jövőbeni fejlesztési lehetőségekre is. A keretrendszer egy különleges proxy szerverből és legalább egy kliensből áll. A keretrendszer ismertetését és üzemeltetését az szerver/kliens moduljaival kezdem.

Weboldalak intelligens szimulálása

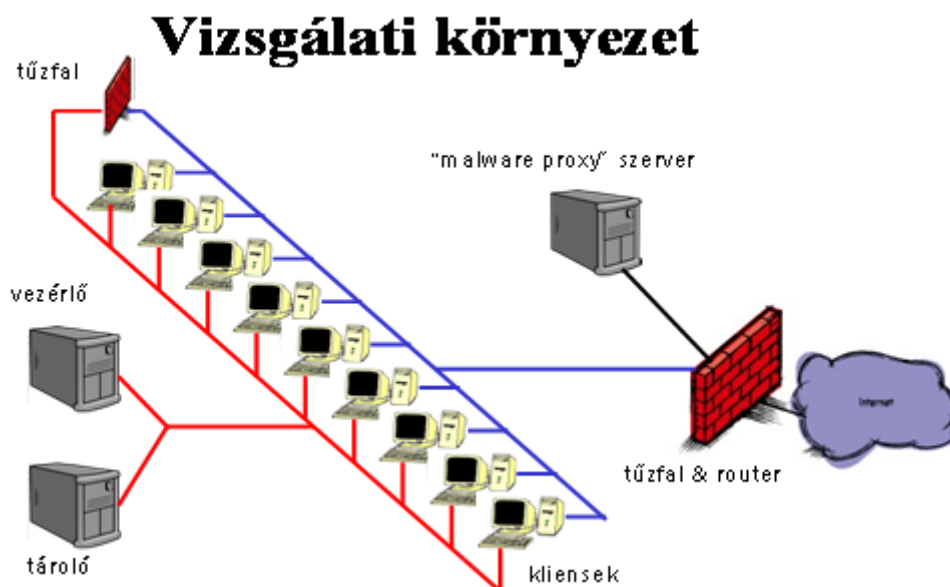
2. Bevezetés

Régen a kártyák floppykon és cd-ken terjedtek, manapság ez teljes mértékben megváltozott és elsőszámú hordozóként az internetet tekinthetjük. A támadás irányulhat egy program ellen vagy ami a statisztikákat tekintve a könnyebb a felhasználó ellen. Ezeknek a kártékony kódoknak az elérhetősége villámgyorsasággal változhat, néha csak pecekig, óráig élnek az url linkek. Ez nagy problémát jelent a detektálásukban illetve az elkapásukban. Erre kívánok megoldást nyújtani rendszeremmel, mellyel a kártékony oldalak, linkek lementhetővé válnak és egy biztonságos tesztelői környezetbe visszajátszható a lementés pillanatában létrejött állapot. A dolgozatomban kitérek a teljes rendszer, illetve az egyes részek működésére, a fejlesztés korai stádiumában fellépet nehézségekre és a jövőbeni fejlesztési lehetőségekre is.

Weboldalak intelligens szimulálása

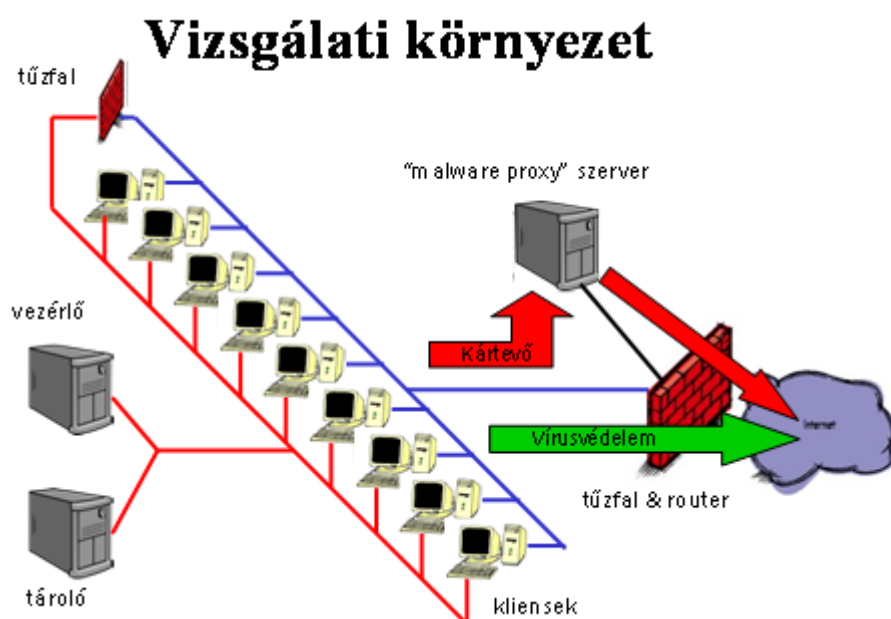
3. A rendszer működési elve és telepítésének lépései

A rendszerhez minimális működéséhez szükséges egy szerver és legalább egy kliens gép. A szerver gép funkcióját tekintve egy különleges proxy szerverként üzemel, mely a magán áthaladó megfigyelni kívánt weboldal http forgalmát monitorozza és menti egy későbbi visszajátszásra. Az egyes feladatok kiosztását és végrehajtásának figyelését a Manager szerver látja el. A szerver gép két hálózati csatlakozóval rendelkezik(eth0 és eth1), melyekből az eth0 csatlakozik a külvilág fele, az eth1 csatlakozik a kliens gépekhez. Fontos tudni, hogy míg az eth0 lehet dhcp áttall konfigurálva addig az eth1 mindig fix ip címmel kell, hogy rendelkezzen. A szerver gép részletes felkonfigurálását és egy kezelését a 3. fejezet tárgyalja. Az alábbi képen a rendszer sematikus ábráját láthatjuk letöltés közben, melyben egy Manager szerver, egy proxy szerver, egy útválasztó és három darab kliens vesz részt. Letöltés közben a kliens gépek elérik az internetet, a külvilágot így képesek meglátogatni a weboldalakat. A proxy eközben a cél weboldal teljes http forgalmát lementi.



Weboldalak intelligens szimulálása

A rendszer működése visszajátszás esetén teljesen másképp működik. Ebben az esetben a kliens gépek számára alap esetben (lásd fejlesztési irányvonalak részt) az internet nem elérhető csak a proxy szervert érhetik el. A szerver ebben az esetben az elmentett állást visszajátssza a hálózatba a megfelelő gépek számára.



A visszajátszás természetesen történhet virtuális gépre vagy adott esetben egy teljesen másik gépkörnyezetben, ilyenkor csak a szerver gép beállításait, konfigurációs állományait és persze a visszajátszani kívánt weboldal mentési mappáját kell vinnünk. A rendszer megtervezésekor törekedtem a szabványkövetésre, hogy a későbbi változások illetve a web fejlődése könnyen és biztonságosan lekövethető legyen.

Weboldalak intelligens szimulálása

3.1 A proxy szerver tisztatelepítése

A rendszer tisztatelepítése esetén szükségünk van egy Debian 5.0 Lenny-t futtató szerverre. Mint előzőekben említettem az egyetlen hardveres megkötés, hogy legalább 2 hálózati csatlakozóval kell, hogy rendelkezzen. Miután feltelepítettük az Operációsrendszert, szükségünk van az alábbi programokra. Megjegyzendő, hogy ha nincs a hálózatban egy gép ami dhcp kiszolgálóként üzemel, lehetőség van arra, hogy ezt a proxy szerver lássa el.

- dhcp3-server
- bind9
- libappconfig-perl
- gawk
- tshark, tcpick
- Perl és moduljai

Ezenfelül szükség van az egyes modulok működéséhez szükséges CPAIN modulok-ra. Ezeknek a telepítése kiemelten fontos mivel ennek elmulasztása a későbbiekben nem azonnal észrevehető hibát okoz így megnehezítve a későbbi esetleges hibakeresést.

A lent látható perl CPAIN modulok verziószámai már biztosan frissültek, ebben az esetben is nyugodtan telepítsük az újabb verziót, mert ebben az esetben is működni fognak.

libappconfig-perl	1.56-2
libcairo-perl	1.060-1
libdbd-mysql-perl	4.007-1
libdbi-perl	1.605-1
libglib-perl	1:1.190-2

Weboldalak intelligens szimulálása

libgnome2-canvas-perl	1.002-1+b2
libgnome2-perl	1.042-1+b1
libgnome2-vfs-perl	1.080-1+b1
libgtk2-perl	1:1.190-1
libhtml-linkextractor-perl	0.130-4
libhtml-parser-perl	3.56-1+lenny1
libhtml-tagset-perl	3.20-2
libhtml-tree-perl	3.23-1
liblocale-gettext-perl	1.05-4
libnet-daemon-perl	0.38-1.1
libnet-dbus-perl	0.33.6-1+b1
libperl5.10	5.10.0-19lenny2
libplrpc-perl	0.2017-1.1
libterm-readkey-perl	2.30-4
libtext-charwidth-perl	0.04-5+b1
libtext-iconv-perl	1.7-1+b1
libtext-wrapi18n-perl	0.06-6
liburi-perl	1.35.dfsg.1-1
libwww-perl	5.813-1
libxml-parser-perl	2.36-1.1+b1
libxml-twig-perl	1:3.32-1
perl	5.10.0-19lenny2
perl-base	5.10.0-19lenny2
perl-modules	5.10.0-19lenny2

Ezek után a dokumentumhoz mellékelt lemeztől a proxy könyvtárat a /root/ könyvtárba kell másolni. A rendszer automatikus működtetése és vezérlése a Manager fejezetben lesz kitárgyalva, a most következő alfejezetek a proxy manuális működtetését és fölkonfigurálást írják le.

Weboldalak intelligens szimulálása

A rendszerben engedélyezett az IP forwarding. Így lehetséges az interfészek között az IP csomagok továbbítása. Ezt a */etc/sysctl.conf*-ban engedélyezhetjük.

```
net.ipv4.ip_forward=1
```

Az itteni beállítás bootoláskor töltődik be. Egy működő rendszer esetében az alábbi fájlok alapján dönthető el, hogy végezze a kernel IP továbbítást.

```
/proc/sys/net/ipv4/ip_forward
```

Ha a fájlban 1 van, akkor engedélyezett.

```
/proc/sys/net/ipv4/conf/$IF/forwarding
```

Ahol **\$IF** egy hálózati interfész neve. (eth0, eth1, tun0, ...)

Ha a fájlban 1 van, akkor adott interfész esetében engedélyezett.

A rendszerben működő szolgáltatások:

- DHCP szerver – eth1-en, belső hálózat számára
- Névszerver – eth1-en, (caching-only módban) névfeloldás a belső gépek számára

A rendszer fejlesztése során igyekeztem a különböző fájl hivatkozásokat relatívan végezni. Így amennyiben a rendszer által használt könyvtárstruktúrát másik helyre mozgatjuk, akkor a szkripteknek ott is működniük kell. A továbbiakban a rendszer helyére a *BASEDIR* segítségével fogok hivatkozni.

Weboldalak intelligens szimulálása

3.1.1. A hálózati konfigurálása

Helyileg a rendszer esetében létezik egy *etc* nevű könyvtár (*BASEDIR/etc*). Itt található a *network.cfg* nevű fájl, amely segítségével tudjuk testre szabni a hálózatunkat. A konfigurációs fájl elején adhatjuk meg az a *networks* direktíva segítségével, hogy mely hálózatok legyenek engedélyezve a definiáltak közül. A *#* segítségével megjegyzéseket tehetünk a konfigurációs fájlba.

```
# Define networks
networks = net0
#networks = net2
```

Szögletes zárójelben definiálhatjuk a hálózat nevét. Ezt követően pedig az adott hálózathoz kapcsolódó paramétereket.

```
[net0]
# network parameters
address = 172.16.1.0
netmask = 255.255.255.0
dhcp_range_start = 172.16.1.10
dhcp_range_end = 172.16.1.254
dhcp_broadcast = 172.16.1.255
dhcp_domain = localdomain
dhcp_dns = 172.16.1.1
dhcp_router = 172.16.1.1
```

A beállítandó paraméterek jelentése a következő:

- *address*: a hálózat címe
- *netmask*: a használandó alhálózati maszk
- *dhcp_range_start*: DHCP 'range' kezdete, innét kezd osztani a DHCP sz.
- *dhcp_range_end*: DHCP 'range' vége, eddig oszt a DHCP sz.
- *dhcp_domain*: DHCP által beállítandó tartomány
- *dhcp_dns*: DHCP által beállítandó névszerver IP címe
- *dhcp_router*: DHCP által beállítandó átjáró

Weboldalak intelligens szimulálása

Konvencionálisan a [net0] hálózat megegyezik az eth1 interfészhez rendelt hálózattal.

[net2]

```
# network parameters
address = 10.0.0.0
netmask = 255.255.0.0
dhcp_range_start = 10.0.0.10
dhcp_range_end = 10.0.255.254
dhcp_broadcast = 10.0.255.255
dhcp_domain = virland.hu
dhcp_dns = 172.16.1.1
dhcp_router = 10.0.0.1
```

Az alábbi minta konfigurációs direktívák segítségével definiálásra kerül kettő hálózat. Viszont a kettő közül csak a net0 hálózat az aktív, mivel a networks = net2 sor kommentezve van. Amennyiben szeretnénk engedélyezni, akkor mindössze annyit kell tennünk, hogy a # -ot kivesszük a sor elejéről.

Ily módon tetszőleges számú hálózat definiálható. Egy dologra kell figyelni, a DHCP szerver esetében korlátozott, a tartomány (range) -beli gépek száma. Így például a 10.0.0.0/8 hálózat használata már problémát jelent a DHCP szervernek. A konfigurációs fájl lehetőséget biztosít arra is, hogy beállítsuk, hogy adott MAC című géphez mindenig ugyanazt az IP címet rendelje hozzá.

[mac]

```
ip = "00:50:56:00:02:01 172.16.1.2 virtest"
```

```
ip = "00:50:56:00:02:02 10.0.5.2 nettest"
```

Weboldalak intelligens szimulálása

A [mac] blokkban az ip direktíva után kell felsorolni a idézőjelben az egyes paramétereket. Az idézőjelben a paraméterek sorrendje a következő (szóközzel elválasztva):

- MAC cím
- IP cím
- hosztnév

Amennyiben az egyes gépeket külön IP címtartományokba helyezzük, úgy nem tudnak egymással kommunikálni. Ha viszont az szeretnénk, hogy lássák egymást, abban az esetben helyezzük őket egy címtartományba .

A hálózat indításához lépünk be a BASEDIR könyvtárba. Ebben a könyvtárban található egy *net.pl* nevű szkript. Ez végzi a hálózat indítását és leállítását az előzőekben bemutatott konfigurációs fájl alapján. A parancsot rendszergazdai szintű felhasználóként kell használni.

A szkript paramétereinek a megtekintése:

```
./net.pl
```

```
usage: ./net.pl { start | stop | configtest }
```

- Hálózat indítása: `./net.pl start`
- Hálózat leállítása: `./net.pl stop`
- A network.cfg szintaktikájának tesztelése `./net.pl configtest`

Az indító szkript a következőket teszi:

- IP alias létrehozása a definiált hálózatok számára
- Az alapértelmezett tűzfal konfiguráció beállítása (BASEDIR/etc/empty)
- DHCP konfiguráció generálás és a szerver indítása

Az leállító szkript a következőket teszi:

- DHCP szerver leállítása
- IP alias-ok törlése
- Az tűzfal konfiguráció törlése, alapértelmezésként minden forgalom mehet

Weboldalak intelligens szimulálása

Az IP alias segítségével egy interfészhez tudunk több IP címet hozzárendelni. Ebben az esetben újabb hálózati interfészek jönnek létre. Mi ezt az eth1-nél akarjuk használni, az interfészek pedig rendre az eth1:0, eth1:1, eth1:2, ... neveket kapják. Ezzel nyílik lehetőség arra, hogy több IP címtartományt használhassunk a belső hálózat esetében. A dhcp3 szerver az alábbi beállításokkal működik.

```
ddns-update-style none;
default-lease-time 3600;
max-lease-time 7200;
authoritative;
log-facility local7;
shared-network nets {
include "/etc/dhcp3/net";
}
```

Az include-al behúzott fájlt (*/etc/dhcp3/net*) hozza létre a *net.pl* szkript. A korábbiakban bemutatott minta konfigurációs fájlok alapján a következő fájl generálódik:

```
subnet 172.16.1.0 netmask 255.255.255.0 {
range 172.16.1.10 172.16.1.254;
option broadcast-address 172.16.1.255;
option domain-name-servers 172.16.1.1;
option domain-name "localdomain";
option routers 172.16.1.1;
}
subnet 10.0.0.0 netmask 255.255.0.0 {
range 10.0.0.10 10.0.255.254;
option broadcast-address 10.0.255.255;
option domain-name-servers 172.16.1.1;
option domain-name "virland.hu";
}
```

Weboldalak intelligens szimulálása

```
option routers 10.0.0.1;
}
host virtest {
hardware ethernet 00:50:56:00:02:01;
fixed-address 172.16.1.2;
}
host nettest {
hardware ethernet 00:50:56:00:02:02;
fixed-address 10.0.5.2;
}
```

A DHCP szerver konfigurációjának megadásakor figyeljünk arra, hogy az alapértelmezett átjáró minden esetben a proxy belső hálózatbeli géphez, adott tartományból hozzárendelt IP cím legyen.

A bind9 konfigurálásakor mindössze a /etc/bind/named.conf.option fájlt kell módosítani. Ide az options blokkon belülre az alábbiakat vettem fel.

```
listen-on { 127.0.0.1; $IPB;};
allow-query { any;};
forwarders { $IPNS; };
```

Ahol

\$IPB: eth1-hez rendelt fix IP cím

\$IPNS. egy név szervernek IP címe, ahova továbbítani kell a kéréseket

A beállítások lehetővé teszik azt, hogy a „belső gépek” számára működjön a névfeloldás.

Weboldalak intelligens szimulálása

3.1.2. A rendszer kezdeti paramétereinek beállítása

A BASEDIR/etc/system.cfg fájl segítségével a rendszer néhány működési paraméterét tudjuk beállítani.

Define system default parameters

gateway = \$GIP

in_ip = \$INIP

capture_min = 5

Ahol

\$GIP: az eth0-hoz rendelt „külső” IP cím

\$INIP: az eth1-hez tartozó „belső” (állandó) IP cím

capture_min: a hálózati forgalom mentés mennyi ideig történjen (percben)

3.1.3. A jelenlegi rendszerhez tartozó fájlok és könyvtárak listája

/net.pl A hálózati indítására, leállítására használható szkript

./system.pl A rendszer adott módú használatára való szkript

/etc

./etc/empty alapértelmezett tűzfal konfiguráció

./etc/network.cfg hálózati konfiguráció

./etc/system.cfg rendszer paraméterek

/log

./log/access_\${ID}.log hozzáférések logolása adott ID-hoz

./log/error_\${ID}.log sikertelen hozzáférések logolása ID-hoz

./log/system_error rendszerhibák naplózása

./net

./net/iptables tűzfal beállító szkript

./net/netup.pl IP alias interfészek indítása, leállítása

./net/readnetconfig.pl network.cfg fájl beolvasása

Weboldalak intelligens szimulálása

`./run` Ide jönnek létre a futás közbeni fájlok

`./run/${ID}` ID-hoz tartozó fájlok

`./run/${ID}.pcap` a hálózati forgalom PCAP formátumban

`./run/db_${ID}` MD5 ellenőrző összegek és fájlok összerendelése

`/system`

`./system/addlearn` „tanulási módban” új tűzfal szabály hozzáadása

`./system/addrun` „futási módban” új tűzfal szabály hozzáadása

`./system/capt_traffic` hálózati forgalom elkapása

`./system/dellearn` „tanulási módban” tűzfal szabály törlése

`./system/delrun` „futási módban” tűzfal szabály törlése

`./system/hreq.pl` egy TCP sessionben lévő HTTP kérések szétválasztása

`./system/hresp.pl` egy TCP sessionben lévő HTTP válaszok szétválasztása

`./system/pick_traffic` TCP kapcsolatok kikeresése a capture fájlból

`./system/readsystemconfig.pl` system.cfg fájl beolvasása

`./system/server.pl` a „szerver funkciót” ellátó perl szkript

`./system/split_http` HTTP kérések, válaszok szétválasztása

Ezen fájlokon kívül a rendszer létrehoz, és használ fájlokat a `/dev/shm` könyvtárban belül.

Ez a könyvtár `tmpfs` -ként van csatlakoztatva, és a memóriában van tárolva. Amit ide elhelyezünk az a gép újraindítását követően törlődik.

Ide a rendszer az alábbi fájlokat hozhatja létre:

`/dev/shm/interfaces` interfészek az IP aliasokhoz

`/dev/shm/net_error` hálózat konfigurálási hiba jelzése

`/dev/shm/state` a rendszer állapota

Weboldalak intelligens szimulálása

3.1.4. A system.pl szkript paraméterei

A rendszer manuális módú indításához lépünk be a BASEDIR könyvtárba. Ebben a könyvtárban található egy *system.pl* nevű szkript. A parancsot rendszergazdai szintű felhasználóként kell használni.

A szkript paramétereinek a megtekintése:

```
./system.pl
```

```
usage: ./system.pl id ip { learn | run | stop | clear }
```

Az id paraméter egyedi azonosításra szolgál. Ennek a segítségével tudunk egy adott állapotot azonosítani.

Az id értéke (\$ID) az alábbi célokra lett felhasználva:

A capture fájl helye: run/\$ID.pcap

A TCP kapcsolatok feldolgozási könyvtára: net/\$ID

A hash adatbázis helye: run/db_\$ID

Tanulás mód indítása:

Ebben a módban létrehoz egy új azonosítójú állapot (id paraméter), a hálózati forgalmat egy fájlba menti el. Ilyenkor történik a „belső gép” megfigyelése, az összes forgalom elkapása.

```
./system.pl 20 172.16.1.2 learn
```

Az konfigurációs fájlban rögzített ideig elmenti az adott IP címről bekövetkezett forgalmat.(20-as azonosító)

Weboldalak intelligens szimulálása

Futási mód indítása:

A tanulás mód sikeresen lefutását követően van arra lehetőségünk, hogy az előállt állapotot „futtassuk”. Ekkor már a belső gépeknek nincsen arra lehetőségük, hogy kifelé kommunikáljanak, csakis a proxy-t érhetik el, illetve a benne tárolt tartalmat. Tegyük fel, hogy van egy 25-ös azonosítóval mentett (és leállított) állapotunk.

```
./system.pl 00000010 10.0.5.3 run
```

Ezzel a paranccsal lehetőségünk van arra, hogy a 10.0.5.3-as IP című gépnek a 00000010-es mentett állapotot adjuk oda. Egy mentett állapot több gép számára is odaadható.

A működésben lévő állapotok megtekintése:

A `cat /dev/shm/state` paranccsal lehetőségünk van arra, hogy megvizsgáljuk az aktuálisan végrehajtás alatt lévő állapotokat.

```
0000010:172.16.1.2:learn
```

```
0000011:10.0.5.2:learn
```

```
0000012:10.0.5.3: run
```

A fájl mezői: id: IP cím: állapot

Tanulási állapotok leállítása:

Jelenleg a `capture_min` eltelését követően még nem történik automatikusan a tanulási állapot megállítása. Ahhoz, hogy adott „megtanult” állapot futtatható legyen, ehhez szükséges az állapot leállítása.

```
./system.pl 00000002 10.0.5.2 stop
```

Emellett természetesen nemcsak „tanulási” állapot, hanem futási állapot is megállítható a korábbi paranccsal.

Weboldalak intelligens szimulálása

Állapot törlése:

Amennyiben egy mentett állapotra már nincsen szükségünk, akkor lehetőségünk van az adott állapot törlésére.

```
./system.pl 00000012 10.0.5.2 clear
```

Ennek hatására az adott azonosítóhoz tartozó fájlok és könyvtárak törlődni fognak, és ezt követően az azonosító újra-felhasználható lesz.

4. A kliens tisztatelepítése és konfigurálás

A kliensgép működését tekintve egy univerzális kompromittálhatatlan web browser, mely az adott weboldalnak megfelelően képes az adott web browser-nek hazudnia magát. Ehhez megint csak a Perl-t vettem segítségül azon is belül az LWP CPAIN modul-t. Miközben a proxy az adott weboldal-kliensgép kommunikációját menti a kliens a célweboldalon rekurzívan végigmegy és bizonyos fájlokat keres, melyek a leginkább veszélyesek lehetnek(a jelenlegi beállítást és annak bővítését ebben a fejezetben tárgyalom ki). Mivel azonban a rendszer nem hajt végre egyetlen fájlt keresés közben így a böngészők illetve mást sebező programok, fájlok nem hamisíthatják meg a mentést.

A kliens gépre szintén egy Debian 5.0 Lenny-t telepítettem, gnome felhasználói felülettel(a visszajátszás esetén szükséges vmware player nem képes a Xwindows-ban való futásra). A mellékleten található DOWN könyvtárat a /root/ könyvtárba kell helyezni. Ezekután a modul futtatásához az alábbi Perl modulokra van szükség.

libappconfig-perl	1.56-2
libcairo-perl	1.060-1
libdbd-mysql-perl	4.007-1
libdbi-perl	1.605-1
libglib-perl	1:1.190-2

Weboldalak intelligens szimulálása

libgnome2-canvas-perl	1.002-1+b2
libgnome2-perl	1.042-1+b1
libgnome2-vfs-perl	1.080-1+b1
libgtk2-perl	1:1.190-1
libhtml-linkextractor-perl	0.130-4
libhtml-parser-perl	3.56-1+lenny1
libhtml-tagset-perl	3.20-2
libhtml-tree-perl	3.23-1
liblocale-gettext-perl	1.05-4
libnet-daemon-perl	0.38-1.1
libnet-dbus-perl	0.33.6-1+b1
libperl5.10	5.10.0-19lenny2
libplrpc-perl	0.2017-1.1
libterm-readkey-perl	2.30-4
libtext-charwidth-perl	0.04-5+b1
libtext-iconv-perl	1.7-1+b1
libtext-wrapi18n-perl	0.06-6
liburi-perl	1.35.dfsg.1-1
libwww-perl	5.813-1
libxml-parser-perl	2.36-1.1+b1
libxml-twig-perl	1:3.32-1
perl	5.10.0-19lenny2
perl-base	5.10.0-19lenny2
perl-modules	5.10.0-19lenny2

A rendszer későbbi fejlesztési szakaszában jelentkezett egy kellemetlen hiba mely miatt hiába képes a rendszer működni új környezetben de ismeretlen hiba üzenetek keletkeztek melyek hosszas keresés alapján a libhtml-parser-perl perl modul frissítéséből eredt, ezért fontos, hogy rendszer frissítés előtt

Weboldalak intelligens szimulálása

lehetőleg nézzük át az érintett modulok dokumentációját az esetleges változásokról. Ezek után már nincs másra szükségünk csak az igényeknek megfelelően konfigurálni.

4.1 A rendszer használata és konfigurálása

A /root/DONW/ könyvtár az alábbi könyvtárstruktúrával rendelkezik.

- /DOWN/agent/ Itt tárolódnak az egyes web böngészőket azonosító useragentstring-ek.
- /DOWN/download/ A letöltött tartalmak a weboldalak ID-vel ellátott(későbbi fejezetben kifejtve) mappastruktúrája.
- /DOWN/logs/ A rendszer működése közben fellépő események naplózása, ezáltal a hibakeresés sokban javul.
- /DOWN/down.cfg A modul hely érzékeny konfigurációs állománya(Fontos, hogy az itt és a később leírt testworker mappában található konfigurációs állomány megegyezzen).
- /DOWN/down.pl
- /DOWN/Linkcollect.pm
- /DOWN/LogWrite.pm
- /DOWN/ReadConfig.pm

Először is tekintsük át a UserAgentString-et tartalmazó mappát. Jelenleg a könyvtárban alapesetben a két .agent található. A rendszer alapesetben a default.agent-et használja. A default.agent az alábbiak szerint néz ki

User-Agent: Mozilla/5.0 (X11; U; Linux i686; hu-HU; rv:1.9.0.19)

Gecko/2010072023 Icedweasel/3.0.6 (Debian-3.0.6-3)

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

Accept-Language: hu-hu,hu;q=0.8,en-us;q=0.5,en;q=0.3

#Accept-Encoding: gzip,deflate

Weboldalak intelligens szimulálása

Accept-Charset: ISO-8859-2,utf-8;q=0.7,*;q=0.7

Keep-Alive: 300

Connection: keep-alive

A UserAgent-String azért fontos, mert hiába a mindenki által elfogadott webes szabványok a web böngészők mindig egy kicsit másképp jelenítik meg a weboldalt a konkurens böngészőkhöz képest. Ebből következik, hogy a különbségek miatt különböző sebezhetőségek is keletkeznek melyeket a kártékony kódokat terjesztő weboldalak ki is használnak. Ezért fontos a különböző UserAgentString használata. Fontos megjegyezni, hogy a rendszer csak azonosításra használja és nem az adott web böngésző szerint látogatja meg az oldalt, ezáltal is biztosítva, hogy ne fussanak le az esetleges kártékony kódok. Jelenleg a default.agent utf-8 karakterkódolásban látogatja meg az oldalakat tehát az ékezetek ennek megfelelően eltűnnek a, ennek csak esztétikai hátránya van, a weboldal káros kódját ez nem érinti, igény esetén lehetséges az egyéni karaktertáblák hozzáfűzése ezáltal lehetőség nyílik az ékezetes és egyéb „egzotikus” weboldalak hű lemásolása. Letöltés közben a log-ba a hibás karakterkódolást is feljegyeztem, hozzáfűzve, hogy milyen karaktertáblát használ eredetileg a weboldal.

A sorban következő fontos állomány a down.cfg. Ebben a fájlban határoztam meg, hogy a rendszer milyen munkakönyvtárakat használjon és azoknak mi a direkt elérési útvonala.

```
# Global parameters
[global]
sysdir = /root/DOWN
syslog = $sysdir/logs
filesdir = $sysdir/download/files
htmldir = $sysdir/download/html
logdir = $sysdir/download/logs
tmpdir = $sysdir/download/tmp
```

Weboldalak intelligens szimulálása

```
agentdir = $sysdir/agent
```

```
# not used
```

```
net_max-retries = 3
```

```
net_timeout = 30
```

Ha esetleg a weboldalt tároló szerver nem válaszolna vagy menetközben törölték volna megadtam a minimális biztonsági limitet, mely megszabja, hogy hányszor és mennyit várakozzon mielőtt egy bontaná a kapcsolatot a weboldallal. Alapesetben ez 3*30 másodperc. Tetszőleges értékkel módosítható.

```
#URI-s to manage recursion
```

```
search_pattern=\.html|\.php|/$
```

Továbbá tartalmazza, hogy milyen webes fájlokat keressen, hogy az azokban található linkek alapján letöltse a teljes weboldalt(ez alapján találja meg a bináris fájlokat is). Meghatunk még bizonyos kiterjesztéseket melyeket a rekurzív keresés során letölt. Ide tetszőleges számú kiterjesztést adhatunk meg. Ezeket a /root/DOWN/download/files helyezi a wboldal+id elnevezéssel. Jelenleg az alábbi fájlokat keresi a rendszer.

```
#extensions for download
```

```
extensions = exe
```

```
extensions = zip
```

```
extensions = msi
```

```
extensions = doc
```

```
extensions = txt
```

```
extensions = pl
```

```
extensions = css
```

```
extensions = bz2
```

```
extensions = gz
```

Weboldalak intelligens szimulálása

Miután a megfelelő beállításokat megtettük manuálisan az alábbi paranccsal tesztelhetjük le a rendszer működőképességét.

usage:

down.pl ID URL [http_headers_file]

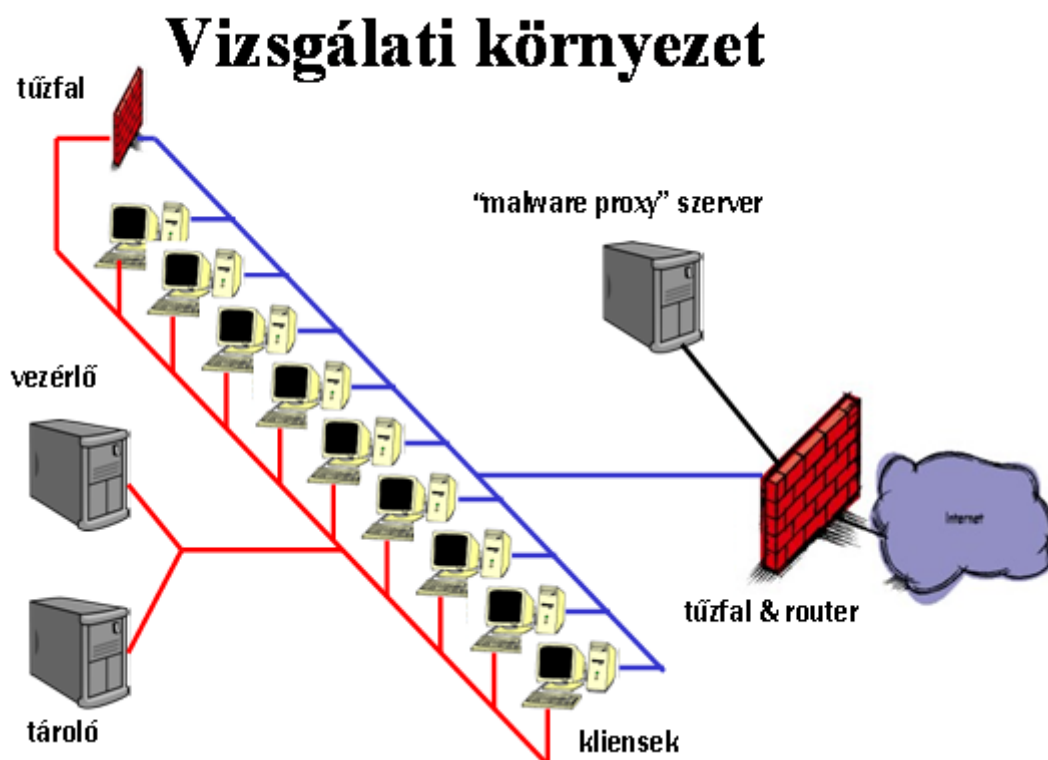
Az ID egy nyolc karakterből álló számsor mely egyedi azonosító és ennek felhasználásával lehet a későbbiekben visszakeresni egy mentet állapotot. Az URL a kártékony weboldal elérési útvonala(fontos megemlíteni, hogy manuális indítás esetén a webcím végén kötelező elhelyezni egy / jelet, mert ez zárja le az url-t). Opcionálisan megadható, hogy az agent könyvtárból melyik agent-et használja. Ez akkor lehet hasznos ha tudjuk, hogy a weboldal egy bizonyos böngészőre(vagy annak valamely típusára) készült. A weboldal lementését egyrészt a Manager modul számára elhelyezett stop fájl jelzi, manuális indítás esetén a prompt-ot kapjuk vissza. Hiba vagy kódlap hiba esetén(lásd a 15. oldalt) az éppen használt tty-n írja ki, természetesen a naplóban is szerepel minden fájl sikeres illetve sikertelen letöltése dátummal ellátva.

5. A vizsgálati környezet strukturális felépítése és a tesztelési módszere, valamint felhasználásának fontossága

A rendszert a fejlesztés későbbi fázisában lett átalakítva, hogy automatikus működésre legyen képes. A Veszprog Kft.-nél használt tenyésztőrendszerbe való integrálhatóság lett célul kitűzve mely kisebb nagyobb hibákat követően sikeresnek mondható.

A tenyésztőrendszerben található egy központi gép mely a feladatok kiosztásáért felel, ezt kellett úgy módosítani, hogy a meglévő rendszer is működőképes maradjon és könnyen lehessen a tenyésztés/MPS használata közt váltani. Ehhez először is egy NFS megosztást kell létrehozni a Manager Szerver-en,, hogy az egyes szereplők kommunikálhassanak és szabadon elérhessék a szükséges fájlokat. Ahhoz, hogy megértsük az automatikus működést látnunk kell a teszt rendszert.

Weboldalak intelligens szimulálása



Minden kliens rendelkezik egy virtuális géppel amin a biztonsági rendszerek tesztelése folyik. A visszajátszási folyamat lehetőség nyílik arra, hogy minden egyes vírusvédelmi szoftvernek pontosan ugyanazt a körülményt alakítsuk ki ezáltal képesek legyünk összehasonlítani az egyes megoldásokat.

Weboldalak intelligens szimulálása

A rendszer egyik fontos és kiemelt funkciója, hogy egyszerre több elmentett állapotot is képes visszajátszani, ezenfelül hiányzó kérések későbbi hozzáfűzése már elmentett állapotokhoz így létrehozva egy komplex valós környezetet.

Egy szokványos elmentett állapot esetében kiemelten fontos, hogy csak a hasznos adat elmentése következzen be amit az alábbi képen észre is vehetünk. Ez egy szokványos már szűrt állapotban elmentett pcap állomány melyben a vizsgálati rendszer és a cél szerver közti kommunikáció forgalma található.

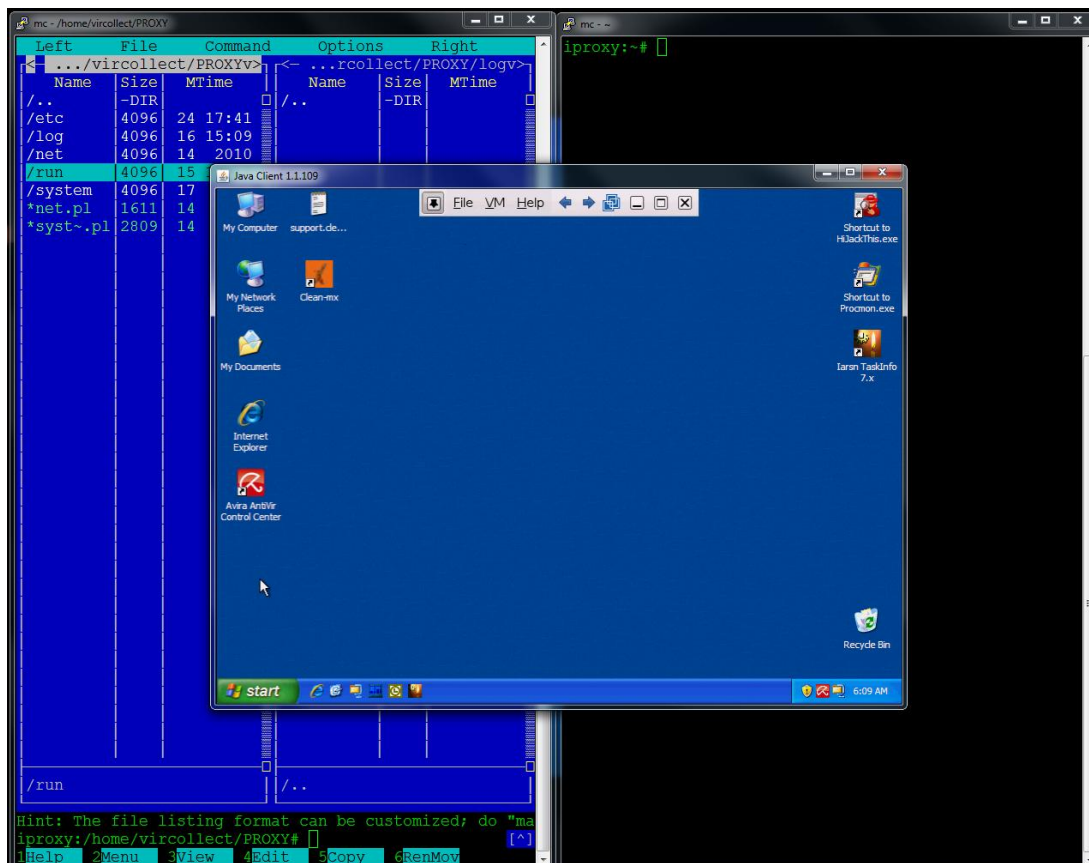
1	0.000000	192.168.68.2	193.225.187.83	TCP	46526 > http [SYN] Seq=0 win=5840 Len=0 MSS=1460 SACK_PERM=1 T
2	0.021496	193.225.187.83	192.168.68.2	TCP	http > 46526 [SYN, ACK] Seq=0 Ack=1 win=5792 Len=0 MSS=1380 SA
3	0.021741	192.168.68.2	193.225.187.83	TCP	46526 > http [ACK] Seq=1 Ack=1 win=5888 Len=0 TSV=155278868 TS
4	0.053587	192.168.68.2	193.225.187.83	HTTP	GET /vircollect HTTP/1.1
5	0.055949	193.225.187.83	192.168.68.2	TCP	http > 46526 [ACK] Seq=1 Ack=415 win=6912 Len=0 TSV=464110356
6	0.056705	193.225.187.83	192.168.68.2	HTTP	HTTP/1.1 301 Moved Permanently (text/html)
7	0.056806	193.225.187.83	192.168.68.2	TCP	http > 46526 [FIN, ACK] Seq=557 Ack=415 win=6912 Len=0 TSV=464
8	0.057034	192.168.68.2	193.225.187.83	TCP	46526 > http [ACK] Seq=415 Ack=557 win=6976 Len=0 TSV=15527887
9	0.058699	192.168.68.2	193.225.187.83	TCP	46526 > http [FIN, ACK] Seq=415 Ack=558 win=6976 Len=0 TSV=155
10	0.060206	192.168.68.2	193.225.187.83	TCP	46527 > http [SYN] Seq=0 win=5840 Len=0 MSS=1460 SACK_PERM=1 T
11	0.069208	193.225.187.83	192.168.68.2	TCP	http > 46526 [ACK] Seq=558 Ack=416 win=6912 Len=0 TSV=46411035
12	0.081477	193.225.187.83	192.168.68.2	TCP	http > 46527 [SYN, ACK] Seq=0 Ack=1 win=5792 Len=0 MSS=1380 SA
13	0.081758	192.168.68.2	193.225.187.83	TCP	46527 > http [ACK] Seq=1 Ack=1 win=5888 Len=0 TSV=155278883 TS
14	0.082074	192.168.68.2	193.225.187.83	HTTP	GET /vircollect/ HTTP/1.1
15	0.086903	193.225.187.83	192.168.68.2	TCP	http > 46527 [ACK] Seq=1 Ack=416 win=6912 Len=0 TSV=464110363

Fontos megjegyezni, hogy a későbbi archiválás miatt a strukturált letárolás nagyon fontos így ennek megfelelően lett kialakítva a rendszer. Az alábbi képen egy káros oldal letárolt felépítését láthatjuk.

	00000042		2011.09.17. 14:02:15	rwxr-xr-x	root
	00000044		2011.10.24. 12:07:11	rwxr-xr-x	root
	00000042.pcap	5 976	2011.09.17. 14:00:51	rw-r--r--	root
	00000043.pcap	193 965	2011.09.17. 14:08:06	rw-r--r--	root
	00000044.pcap	20 004	2011.10.24. 11:59:14	rw-r--r--	root
	db_00000042	680	2011.09.17. 14:02:26	rw-r--r--	root

Weboldalak intelligens szimulálása

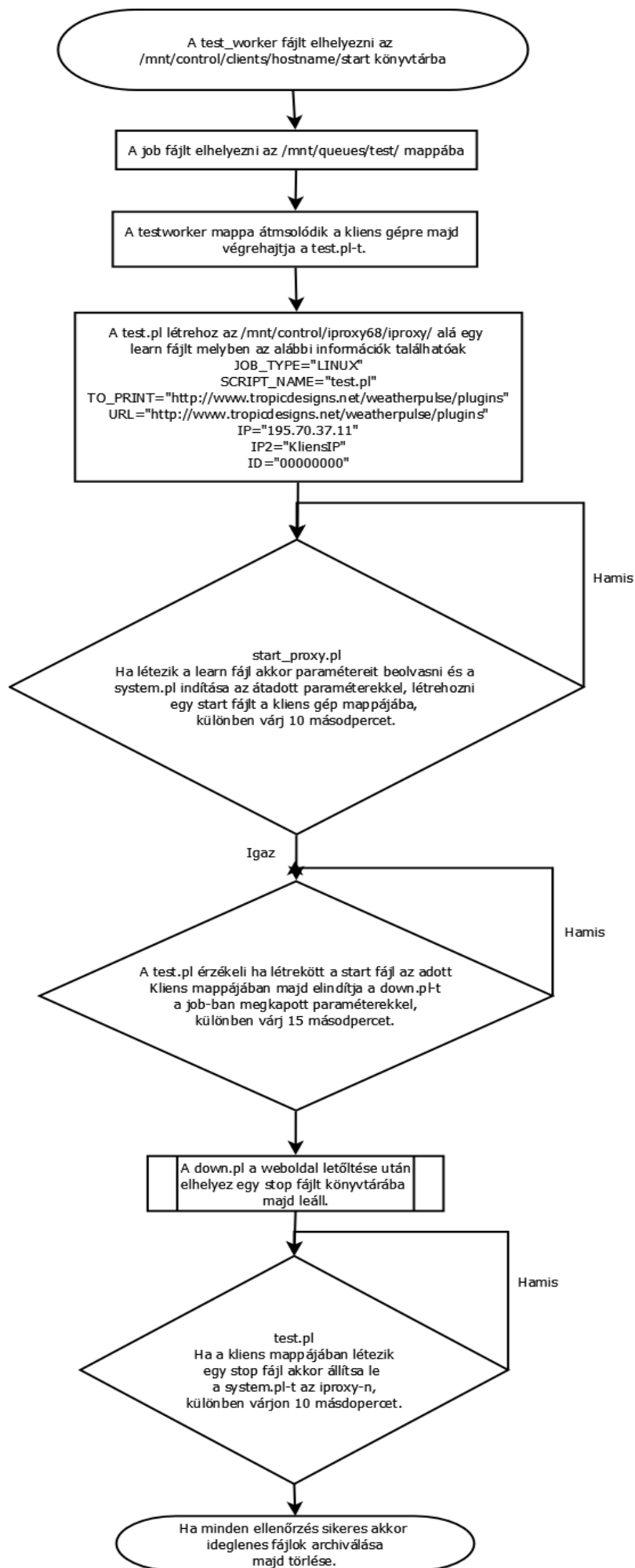
Itt láthatjuk az elmentett adatforgalomból kiszűrt hasznos kérés/válasz kommunikációt valamint az erről készített adatbázist. Ez azért fontos, hogy a rendszer időrendi sorrendbe tudja tenni az egyes kéréseket és válaszokat. Ezek egy az ID-val ellátott mappában kerülnek mentésre. Itt lesznek továbbá elmentve az egyes HTML, php állományok melyek a weboldal szerves részét képezik. Ezeknek a fájloknak a száma és kiterjesztése attól függ, hogy milyen fájlokat tartalmazott a weboldal az adott időpillanatban. Minden egyes fájlról mely letöltésre kerül készül egy olyan adatbázis, mely tartalmazza az md5, sha1 lenyomatát így biztosítva, hogy ne lehessen kompromittálni az elmentett állapotot és hiteles másolat legyen, valamint az egyes adat degenerációk felismerését teszik lehetővé. A lent látható képen a vizsgálati rendszert és a hozzá szervesen kapcsolódó virtuális környezetet láthatjuk melyben az egyes biztonsági megoldások tesztelése folyik. Erre a virtuális rendszerre kerül kiküldésre az elmentett állapot.



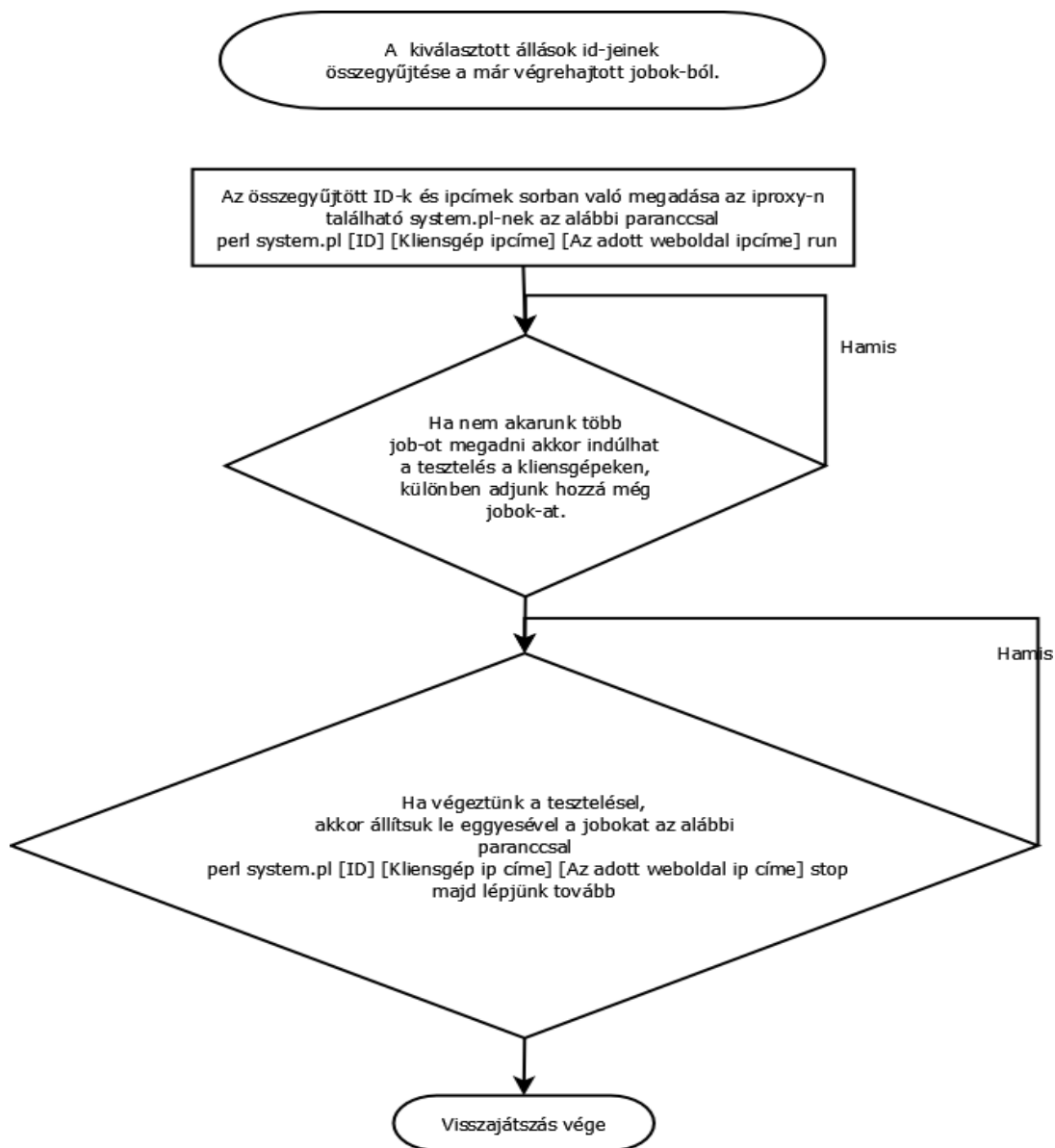
Weboldalak intelligens szimulálása

5.1 A vizsgálati környezet automatikus működésének folyamatábrája

A rendszer működését, nehéz lenne átlátni ha fejezetekre bontanám így egy lépéssorozatban magyarázom el, hogy mi történik egy letöltés esetén illetve visszajátszás során, kitérek az egyes vezérlőmodulok működésére, telepítésére illetve az esetleges hibákra. Mielőtt elkezdeném a részletes bemutatást érdemes áttekinteni a rendszer működését automatikus indítás esetén.



Weboldalak intelligens szimulálása



Visszajátszás esetén kézzel kell végrehajtanunk az alábbi folyamatot, mely két parancsból áll. Az alábbi folyamatára segít megmutatni ezt az egyszerű és könnyen megjegyezhető folyamatot.

Weboldalak intelligens szimulálása

6. A rendszer fejlesztetősége és jelenlegi fejlesztései

Mivel e rendszer működése nagyban függ a jelenlegi és jövőbeni webes illetve támadói metodikák fejlődésétől belátható, hogy a fejlesztés sohasem állhat le, hogy mindig hiteles másolatok készülhessenek a legrövidebb idő alatt az adott weboldalról illetve kártékony kódokról. Későbbiekben nagyon fontosnak tartom, hogy a visszajátszást is automatizáljam azáltal, hogy egy intelligens modul folyamatosan figyeli a vizsgálati rendszert és egy kérés esetén kiszolgálja azt a belső tárból vagy adott estében egy rögzítéssel új tartalmat vegyen fel ezáltal komplexebben vizsgálva a kártevők működését.

Weboldalak intelligens szimulálása

7. Összefoglalás

Manapság a legtöbb kártevő kártékony URL-ken, kártékony oldalakon keresztül terjed. Ezeknek az oldalaknak a tartalma folyamatosan változik: megjelennek, eltűnnek, esetleg megváltozik a tartalmuk. A kártevők aktivitását így nagyon nehéz követni, esetenként csak néhány hétig, napig vagy óráig elérhetőek.

A malware proxy server képes arra, hogy weboldalak tartalmát archiválja és a forgalmat egy későbbi időpontban megismételje. A “malware proxy server” módszere egyszerűbbé és könnyebbé teszi a kártevők elemzését, a védelem fejlesztését, illetve a védelmi rendszerek vizsgálatát.

Weboldalak intelligens szimulálása

8. Felhasznált irodalom

Dr. Leitold Ferenc, Horváth Botond – Dobozba zárt internet, Hacktivity 2011, Budapest

Ször Péter – A vírusvédelem művészete SZAK KIADÓ KFT. 2010

Pere László - GNU/Linux rendszerek üzemeltetése II. 2005

Andrew S. Tanenbaum - Számítógép-hálózatok (új, bővített kiadás - Panem) 2003

Peter Norton – A hálózat biztonság alapjairól Kiskapu 2000

Leitold Ferenc: Automatikus vírusanalizáló rendszer, Proceeding of the HISEC'96 Conference, Budapest, pp. 112-119, 1996

Leitold Ferenc: Automatic Virus Analyser System, Proceeding of the 5th International Virus Bulletin Conference, Boston, USA, pp. 99-107, 1995

<http://www.buhera.blog.hu>

<http://www.hup.hu>

<http://msdn.microsoft.com/en-us/ms348103>

<http://www.securitytube.net/>