6. fejezet - Tartalom

6.1. A szkenner

6.2. Szkennelesi eljárások

6.2.1 Szkenneles teljes TCP-kapcsolattal

6.2.2 Félig nyitott TCP-szkennelés

6.2.3 TCP-FIN-Scan

6.2.4 Fragmentált szkennelés

6.2.5 UDP-ICMP-Port-Unreachable szkennelés

6.2.6 UDP-Recvfrom-And-Write szkennelés

6.2.7 ICMP-echo-scanning/ping-szkennelés

6.2.8 A nyitott portok csak a kezdetet jelentik

6.2.9 A portszkennelés elleni védelem

6 Szkennelés - A rések keresése

A *szkennelés* az egyik legfontosabb módszer egy rendszer gyenge pontjainak a felismerésére. No persze nem a hagyományos értelemben vett szkennelésről van szó, hiszen e szó jelentés itt az, hogy egy rendszer minél több csatolófelületét, illetve portját szólítják meg, hogy kitalálják, melyek engedik meg kapcsolat felépítését.

A *portszkennek* a támadók legfontosabb eszközei. Minden hacker elvégzi a portszkennelést a célrendszerén, mielőtt munkához látna. Emellett a portszkennelés arra is szolgál, hogy a rendszergazdák ellenőrizzék rendszerük biztonságát. Ezért minden rendszergazdának jól kell ismernie a szkennereket.

A következő magyarázatok feltételezik a közkedvelt protokollok alapismereteit, különben nehezen érthetők az összefüggések. Ehhez minden fontos információ megtalálható a 3. fejezetben.

6.1. A szkenner

A szkenneléshez szükség van egy *portszkennerre.* A legismertebb szkennerek közé tartozik a *Nessus* és az *nmap* - mindkettő a Linux alatt gyökerezik, de időközben már Windowshoz is kifejlesztették. A szkennerek kérdéseket küldenek a portoknak, és visszadják a rendszer válaszait. Közben különböző információkat gyűjtenek, többek között arról, hogy milyen szolgáltatások futnak a rendszeren, lehetségesek-e anonim loginek, és hogy megkövetelik-e a hitelesítést a hálózati szolgáltatások.

6.2 Szkennelési eljárások

Egy rendszer szkennelésére különböző lehetőségek kínálkoznak. Hogy ezek közül melyiket választják, az azoktól a feltételektől függ, amelyek mellett a feladatot meg kell oldani. A rendszer minden jogosultságával rendelkező rendszergazda teljes áttekintést szeretne kapni, lehetőleg gyorsan, a hacker pedig észrevétlen szeretne maradni.

6.2.1 Szkennelés teljes TCP-kapcsolattal

Itt egy olyan szkennelési eljárásról van szó, amelynek a célja *TCP alapú szervizeket/szolgáltatásokat találni.* Ehhez a következőképpen járnak el: a szkenner, illetve a kliens megpróbál teljes kapcsolatot felépíteni a host minden portjához. Ha a szkenner ezeknél a kísérleteknél egy porttól feleletet kap, akkor az *elérhetőnek* számít. A szkenner egy TCP/IP csomagot küld SYN-flaggel a számítógépnek (a flagek [zászlók] speciális ismertetőjelek a csomag TCP-headerében), a host visszajelzése pedig, ha nem érhető el a port, *RST-ACK,* ha elérhető, akkor *SYN-ACK.* Ebben az esetben a szkenner ismét *ACK-val* válaszol, és a kapcsolat létrejött.

Ezen a módon persze nagyon sokáig tartana minden kapcsolatot felépíteni és azután befejezni, és kérdéses lenne, hogy minden portot le lehet-e szkennelni. Az mindenesetre biztos, hogy minden időkeretet túllépne. Éppen ezért a szkenner *multi-socket eljárást* alkalmaz, amelynél egyidejűleg nagyon sok kérdést tud feltenni.

Ennek az eljárásnak az előnye, hogy nagyon gyorsan vezet eredményre. A támadóknak azonban nem felel meg, mert nagyon feltűnő, és nagy valószínűséggel a támadó IP-címe is beíródna a rendszer logfájljaiba.

6.2.2 Félig nyitott TCP-szkennelés

A félig nyitott szkennelésnél nem állítanak elő teljes kapcsolatot a hosthoz, hanem a *SYN-ACK* megerősítés után egy elérhető sockettől *RST-ACK-val* rögtön megszakítják a kapcsolatot. Ha a szkennelt számítógép *biztonsági és felügyeleti státusza* csekély, akkor nem fogják észrevenni ezt a szkennelést. Az előnye viszont, hogy Unix alatt csak az tud végrehajtani ilyen eljárást, akinek root-jogai

vannak. Ez gondoskodik ugyanis arról, hogy a hálózatokban az erre jogosult rendszeradminisztrátoron kívül senki se tudjon végrehajtani ilyen szkennelést alacsonyabb jogokkal.

6.2.3 TCP-FIN-Scan

A *TCP-FIN* szkennelést a Unix alatti TCP-implementációk többségének egy hibája (bug) teszi lehetővé. A szkennelendő portra egy TCP-csomagot küldenek egy FIN-csomaggal. A FIN-csomagokat arra használják, hogy a fennálló kapcsolatot annak rendje és módja szerint bezárják. A lezárt portok RST-csomaggal válaszolnak a FIN-csomagra, a nyitottak ezt nem teszik, ezért lehet a TCP-FIN szkennel könnyen kideríteni a nyitott portokat. A szkennelésnek ez a módja azok közé a csekély számú eljárások közé tartozik, amelyek nem feltűnőek. Ezek a „letapogatások" azonban csak Unix alatt működnek, a Microsoft operációs rendszereknél a TCP-implementációnak nincs ilyen fajta hibája.

6.2.4 Fragmentált szkennelés

A *fragmentált szkennelés* a TCP-szkennelések egy további módja. Ennél a TCP-csomagok több kis csomagra fragmentálódnak, illetve darabolódnak. Ezen a módon lehet meggyőződni arról, hogy nem fogja-e egy tűzfal felfedezni a portszkennelést.

6.2.5 UDP-ICMP-Port-Unreachable szkennelés

Az *UDP,* más protokollokkal ellentétben, nem igazolja vissza a fogadást. Csak a lezárt port küld vissza egy üzenetet, hogy a port nem elérhető. Tehát szkenneljük a portokat, és várunk egy UDP-ICMP-PORT-UNREACH üzenetre. A szkennelésnek ez a módja nagyon hosszadalmas, és a pontossága nagy mértékben a szkennelt számítógép kihasználtságától, illetve rendszererőforrásaitól függ.

Ráadásul csak Linux alatt működik, és azok a felhasználók, akik rootként vannak bejelentkezve, megkapják az ICMP PORT UNREACH üzeneteket. Ezeket a szkenneléseket csak a rendszergazda tudja elvégezni, hiszen *system admin* jogok kellenek hozzá.

6.2.6 UDP-Recvfrom-And-Write szkennelés

Ellentétben a UDP-ICMP-PORT-UNREACH szkenneléssel, amelynél csak azok a userek kapnak pozitív visszajelzést, akik rootként vannak bejelentkezve, a *UDP-Recvfrom-And-Write* szkennelés egy normál módon bejelentkezett felhasználónak is lehetővé teszi, hogy „érdekes" jelzéseket kapjon.

A háttérben ismét egy bug áll: ha megpróbálunk egy portra írni, amelyik a UDP-ICMP-PORT-UNREACH szkennelésre ICMP-PORT-UNREACH választ adott (amiről normál felhasználóként nem értesülünk), akkor többnyire ezt az üzenetet kapjuk: *Error 13 - Try Again,* a normál *Error 111 - Connection refused* üzenet helyett. Ennél az eljárásnál tehát minden portot kétszer szkennelnek, egyszer, hogy a számítógép kiadjon egy választ, amit sajnos, nem látunk, és másodszor, hogy mégis kapjunk informatív visszajelzést - 13-as hibánál a port le van zárva. Természetesen ez az eljárás is nagyon időigényes, és gyakran megbízhatatlan.

6.2.7 ICMP-echo-scanning/ping-szkennelés

A szkennelésnek ez a módja csak azt állapítja meg, hogy melyik hostok elérhetők. Ez azt jelenti, hogy egy *ICMP echo scanning/ping* szkennelést hajtanak végre, hogy találjanak egy hostot, mielőtt még egy megtalált rendszeren el lehetne indítani a tulajdonképpeni portszkennelést. Ehhez egy ICMP-ECHO-csomagot vagy egyszerűen egy pinget küldenek kölönböző IP-címekre. Ha a számok valamelyike mögött van egy rendszer, az ICMP-ECHO-REPLY csomaggal válaszol. Megvan tehát a cél, amelyen a támadó a legkülönbözőbb szkennelésekkel réseket kereshet.

6.2.8 A nyitott portok csak a kezdetet jelentik

Ha találtatott egy rendszer nyitott portokkal, akkor egy portlista segítségével lehet áttekintést kapni arról, hogy milyen szervizek (FTP, Telnet, HTTP, HTTPS, stb.) futnak a rendszeren.

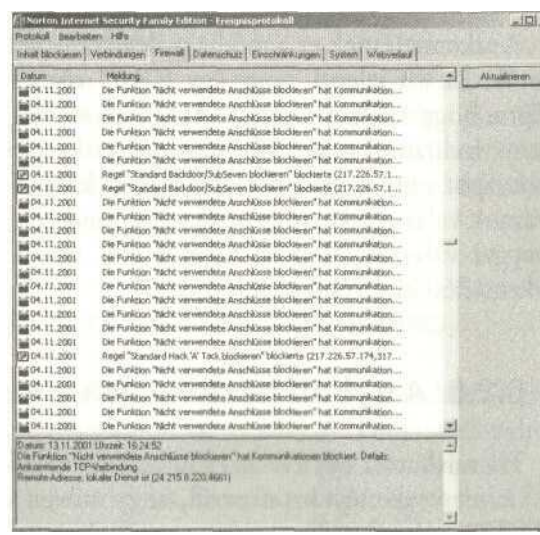
A támadók most hozzákezdenének információkat gyűjteni a célrendszerről. Ez az operációs rendszernél kezdődik, amely a számítógépen fut. Aszerint, hogy milyen portok nyitottak és milyen operációs rendszer van telepítve, a támadó utánanéz a megfelelő kihasználási lehetőségeknek.

6.2.9 A portszkennelés elleni védelem

A szkennelés többnyire az első lépés, amelyet a betörők végrehajtanak, hogy megtalálják és analizálják a célrendszerüket. Ezen keresztül számos fontos információt begyűjthetnek, amelyek nélkül a betörés lehetetlen volna. Ezért fontos védekezni a portszkennelés ellen, illetve naplóztatni a szkenneléseket, hogy azonosítani lehessen a lehetséges támadót, mielőtt még hozzákezdene a tulajdonképpeni támadáshoz.

Természetesen nagyon nehéz valakit visszatartani attól, hogy egy rendszert szkenneljen. A legjobb védelem, *ha a szkennelés nem talál nyitott portokat.* Tehát minden szolgáltatást, ameddig lehetséges, *inaktívan* kellene tartani, hogy ne kínáljunk támadási felületet. Ez azonban többnyire nehezen kivitelezhető, már akkor is, ha valaki például egy FTP-szervert működtet a rendszerén, vagy egy csatlakoztatott hálózat miatt biztosítania kell a 139. port elérhetőségét (lásd. a Windows-megosztásokat a Windows-rendszerek gyenge pontjairól szóló fejezetben). Itt a tűzfalak segítenek, mint amilyen *a Norton Internet Security,* vagy portscan detektorok, mint például a *BlackICE.*

Naplóz a Norton Internet Security



A tűzfalaknak olyan protokoll-funkcióik vannak, amelyek feljegyzik a portszkenneket. A tűzfalakat arra is be lehet állítani, hogy figyelmeztető üzenetet küldjenek, amint TCP-kapcsolatok érkeznek be, és megpróbálkoznak egy portra kapcsolódni.