Capitolul 10

Securitatea reţelelor wireless

10.1 Prezentare generală

Putem considera un atac ca o încercare a unei entităţi neautorizate de a pătrunde în interiorul unui anumit sistem. Atacurile sunt lansate speculând vulnerabilităţi ale sistemului şi având diverse scopuri (sublinierea unor erori de construcţie, atacuri maliţioase, utilizarea resurselor sistemului, pregătirea unor atacuri ulterioare). Pentru atingerea acestor deziderate, un atacator poate efectua atacul de mai multe ori sau poate lansa atacuri combinate plecate de la surse diferite.

Oricum, o clasificare a atacatorilor este dificil de realizat, aceştia dispunând de o paletă largă de motivaţii, experienţă şi obiective. Pot fi până la 26 tipuri diferite de atacatori. O clasificare mult mai simplă este propusă în [3]:

1. **Începători (Script Kiddie)**:

   Sunt în general tineri şi dornici de afirmare, pentru care tentaţia de a se lăuda cu spargerea unor sisteme este mai importantă decât riscurile asumate. Deşi ştiu foarte puţin despre reţelele atacate, inventivitatea şi spiritul de observaţie îi fac să descopere vulnerabilităţi şi astfel să posede un bagaj de cunoştinţe mai larg decât utilizatorii obişnuiţi. Marea lor majoritate se opresc după câteva încercări; totuşi puţini dintre ei, deosebit de dotaţi, se pot ridica la nivelul următor.

   În general începătorii nu fac deosebire între sistemele pe care le atacă. Uzual, ei preferă să construiască o secvenţă de atac pe care o îmbunătăţesc treptat şi o aplică de mai multe ori pentru a vedea ce resurse sunt vulnerabile. Majoritatea lor se pot încadra în genul maliţios: atacă ceva care există, fără a fi interesaţi să obţină foloase deosebite.

2. **Crackeri**:

   Sunt adversari mai experimentaţi şi deci mai periculoşi. Se disting de începători prin abilitatea de a genera atacuri cu strategii bine definite asupra unor ţinte specifice. Uzual, după 1985 ei sunt înglobaţi sub termenul "*hackeri*". În literatură (Jargon File: $http://www.jargon.org$) cei doi termeni sunt diferenţiaţi astfel:

   Definiţia 10.1. *Un* **hacker** *este o persoană care explorează detaliile sistemelor de programare căutând slăbiciuni pe care să le exploateze (spre deosebire de utilizatorii obişnuiţi care preferă să înveţe numai minimul necesar).*

   Un **cracker** *este o persoană care sparge securitatea unui sistem.*

3. **Elite**:

Reprezintă crema celor care pot sparge securitatea unui sistem. În general aici se grupează specialişti guvernamentali, terorişti, radicali politici. Sunt persoane despre care nu se aude aproape deloc şi sunt bine plătiţi sau fanatici.

10.2 Tipuri de vulnerabilităţi

Orice reţea conţine în mod inerent vulnerabilităţi; important este să depistăm cum/când/ unde/ de ce apar ele şi – eventual – cum le putem controla.

La un nivel superior, ele se pot clasifica în:

10.2.1 Vulnerabilităţi software

Diverse studii (academice sau de software-engineering) folosesc ca o măsură a acurateţii unui soft rata de erori găsită la fiecare 1000 linii de cod. Se consideră că majoritatea softului produs în prezent conţine între 5 şi 15 erori la 1000 linii de cod.

Cum sistemele de operare şi aplicaţiile conţin milioane de linii de cod (de exemplu, Microsoft Windows XP conţine cam 50 milioane), chiar şi în cazul în care doar o mică parte din aceste erori sunt legate de securitatea sistemului şi numai o parte din ele pot fi exploatate de intruşi, se ajunge la mii de falii de securitate posibile într-un sistem actual.

În plus, codurile scrise pentru un software se modifică permanent. Uneori, repararea unei slăbiciuni poate genera probleme în altă parte. De asemenea, două părţi independente de soft pot fi sigure dacă sunt luate separat, dar prin rularea lor împreună în cadrul unui sistem pot cauza noi falii de securitate.

Există liste de discuţii dedicate semnalării unor astfel de probleme (de ex. *BugTraq*) care arată cât de multe slăbiciuni se află în softurile actuale şi în cât de multe produse de pe piaţă se află acestea.

10.2.2 Vulnerabilităţi hardware

Acest tip de vulnerabilitate este mai puţin obişanuit, dar importanţa sa creşte datorită numărului tot mai mare de hard programabil aflat pe piaţă (telefoane mobile, tablete etc). Vulnerabilităţi în sistemul intrare/ieşire ($BIOS$), procesorul de reţea sau CPU pot crea pagube mari, deoarece o vulnerabilitate hardware este mult mai dificil de corectat decât una software.

Ca un exemplu, deşi nu a specificat nici o legătură cu securitatea, Intel a oferit utilizatorilor în 1994 înlocuirea gratuită a procesoarelor Pentium din cauza unei erori în calculul cu virgulă mobilă.

10.2.3 Vulnerabilităţi de configurare

Administratorii de reţea fac – cu cele mai bune intenţii – numeroase microconfigurări în reţelele pe care le gestionează. Într-un firewall cu o politică de control acces mai complexă pot există sute de comenzi care permit sau interzic diferite tipuri de trafic. Deci sunt şanse mari ca unele din ele să intre în conflict.

În domeniu mai există şi o problemă numită prescurtat $RFTM$ (Read The Fxxxing Manual). Ea exprimă frustrarea administratorilor de a se confrunta frecvent cu întrebări şi nelămuriri care sunt explicate detaliat în manualele de utilizare. Deci, dacă persoanele responsabile cu dezvoltarea unei tehnologii nu ştiu prea multe despre acea tehnologie, şansele ca ea să lucreze conform standardelor scade semnificativ. Din acest motiv este necesar ca companiile să aloce resurse importante pentru cursuri de specializare.

Conform cu [3], o metodă foarte simplă de a elimina posibile erori de configurare este de a adopta o tehnologie de securitate cât mai simplu de gestionat.

10.2.4 Vulnerabilităţi de politică

O alegere neinspirată a unei politici de securitate (Security Policy) poate permite lansarea unui atac. În general se spune că securitatea unei reţele este la fel de sigură ca politica de securitate pe care o adoptă. Din acest motiv există un proces iterativ prin care securitatea unui sistem este îmbunătăţită în timp prin modificări ale sistemului şi ale politicilor.

Atacurile *DDoS* (care vor fi prezentate mai târziu) sunt exemple de vulnerabilităţi ale politicilor de securitate. La momentul respectiv (anul 2000) astfel de atacuri nu erau nici măcar imaginate. Astăzi frecvenţa lor a generat diverse standarde şi indicaţii pentru a proteja reţelele în faţa atacurilor *DDoS*.

10.2.5 Vulnerabilităţi de exploatare

Faptul că o reţea poate fi exploatată într-un mod sigur nu înseamnă că un utilizator o va utiliza în mod sigur. Vulnerabilităţile de exploatare apar când un utilizator – de obicei începător – violează politica de securitate şi crează o breşă.

De exemplu, utilizatorul adaugă un modem la calculatorul său de servici şi se poate conecta la el înafara orelor de program. În mod normal el instalează personal pe calculator softul respectiv şi deci nu stabileşte o politică de securitate (la care nici nu are acces de altfel). Aşa că un intrus poate folosi acest modem ca punct de lansare al unui atac asupra întregii reţele.

10.3 Rezultate ale atacurilor

Toate atacurile sunt lansate pentru obţinerea unor anumite rezultate. Conform cu [5] sunt patru tipuri de rezultate posibile. Ulterior a fost adăugat un al cincilea. Acestea sunt (în ordinea indicată de autori):

10.3.1 Dezvăluirea informaţiei

Semnifică diseminarea informaţiei oricărei persoane neautorizate de a avea acces la informaţia respectivă. Aici sunt incluse furtul de parole, citirea unor componente de pe hard unde accesul nu este permis, dezvăluirea de informaţii confidenţiale etc.

10.3.2 Coruperea informaţiei

Orice alterare neautorizată a fişierelor stocate sau a datelor transmise prin reţea constituie o corupere de informaţii. Printre numeroasele atacuri de acest gen amintim man-in-the-middle, viruşi care distrug datele, înlocuirea de site-uri web.

10.3.3 Respingerea serviciului

Denial of Service (*DoS*) este degradarea intenţionată sau blocarea resurselor unei reţele sau calculator. Multe tipuri de atacuri flood au ca obiectiv un atac *DoS*, ca şi spargerea intenţioanată a unei reţele pentru a avea acces la resursele sale cu care ulterior se va lansa atacul *DDoS*.

10.3.4 Furtul serviciului

Furtul de servicii este utilizarea neautorizată a calculatorului sau reţelei fără a degrada serviciile celorlalţi utilizatori. Furtul unei parole şi logarea cu aceasta la reţea este un exemplu tipic.

10.3.5 Mărirea accesului

Mărirea accesului (Increased access) este rezultatul unei extinderi a privilegiilor unui utilizator când apelează la serviciile unei reţele. El precede de fapt cele patru atacuri anterioare.

Exemplul tipic este *buffer flow attack* care duce la creşterea accesului neautorizat.

10.4 Taxonomia unui atac

Sub acest termen sunt cuprinse principalele tipuri de atac contra reţelelor precum şi consecinţele pe care le pot avea acestea. Fără o înţelegere a taxonomiei unui atac nu se poate stabili dacă arhitectura de securitate a reţelei este conformă cu politica de securitate stabilită. Principalele clase de atacuri sunt:

- **Citire**: Se poate obţine acces la informaţia neautorizată.
- **Manipulare**: Se poate prelucra/manipula informaţia.
- **Spoof**: Se oferă sau se răspunde cu informaţii false.
- **Flood**: Se blochează resursele sistemului.
- **Redirect**: Se modifică fluxul de informaţii.
- **Composite**: Combinaţie între clasele anterioare.

Fiecare clasă de atac cuprinde mai multe subclase şi/sau atacuri efective; o subclasă cuprinde cel puţin două atacuri efective.

Pentru fiecare atac posibil vom lua în considerare următoarele (a se urmări şi exemplul):

1. **Membru al clasei/subclasei**: Se referă la clasa/subclasa căruia îi aparţine atacul respectiv.
2. **Exemplu de implementări**: Oferă exemple ale atacului asupra unei reţele, website, computer etc.
3. **Preliminarii**: Lista atacurilor care premerg sau urmează atacului în cauză. Astfel, în clasa de atacuri **Citire**, atacul de tip "*Baleierea datelor*" va stabili domeniul adreselor IP pe care ulterior intrusul le va utiliza în lansarea unui atac "*Verif/scan*".
4. **Vulnerabilităţi**: Lista celor mai comune vulnerabilităţi pe care le urmăreşte atacul (din lista celor cinci clase listate mai sus).
5. **Utilizare tipică**: Descrie cea mai obişnuită apariţie a atacului.
6. **Rezultate atac**: Nominalizează cea mai utilizată clasă (din cele cinci) căreia îi aparţine atacul.

7. **Posibile atacuri ulterioare**: Listează cele mai probabile atacuri care pot urma acestui tip de atac.
8. *OSI* **Layers**: O listă a celor mai comune nivele din *OSI* (Open Systems Interconnection) folosite în atac.
9. **Detectare**: O listă a tehnologiilor de securitate capabile să detecteze atacul.
10. **Protecţie**: O listă a tehnologiilor de securitate care pot opri atacul. Ele pot fi utilizate şi la detectare dar nu vor fi cuprinse în ambele categorii.
11. **Dificultatea detecţiei**.
12. **Facilitate de atac**.
13. **Frecvenţă**.
14. **Impact**.
15. **Clasificare generală**.

Ultimele cinci criterii sunt valori numerice (primele patru în intervalul $[1, 5]$ ordonate crescător după gradul de periculozitate. Clasificarea generală se calculează pe o scală de la 1 la 5 derivată din formula

$$(15) = (11) \times 1 + (12) \times 2 + (13) \times 3 + (14) \times 4$$

(s-a notat (x) valoarea de la criteriul x) şi produce o valoare între 10 (introducerea unor mesaje arbitrare în reţea cu speracnţa că vor avea efect) şi 50 (atac distrugător care elimină complet reţeaua).

Exemplul 10.1. *([3]) Să considerăm atacul "Verificare şi scanare)" (prezentat în pag. 8). Atunci:*

1. **Membru al clasei/subclasei***: Citire/Recunoaştere.*
2. **Exemplu de implementări***: Nmap (http : //www.insecure.org/nmap), Nessus (http : //www.nessus.org)*
3. **Preliminarii***: "Baleierea datelor".*
4. **Vulnerabilităţi***: Niciuna (o oarecare formă de verificare sau de scanare asupra IP-urilor din reţele este întotdeauna posibilă).*
5. **Utilizare tipică***: Acces la IP-ul şi aplicaţiiile utilizate de victima atacului.*
6. **Rezultate atac***: Dezvăluirea informaţiei..*
7. **Posibile atacuri ulterioare***: Aproape oricare (după un atac "Verif/Scan" poate urma aproape orice atac). Pe Internet manipulări ale aplicaţiilor sunt foarte obişnuite odată ce a fost scanat un sistem vulnerabil.*
8. *OSI* **Layers***: $3 - 7$.*
9. **Detectare***: IDS (Intrusion Detection System), capabil să detecteze numeroase tipuri de scanare şi examinare) şi firewalls (cu componenta log analysis).*
10. **Protecţie***: Niciuna.*

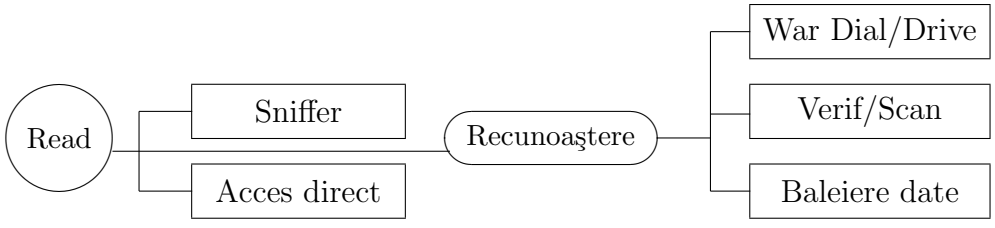
11. **Dificultatea detecţiei**: 4
12. **Facilitate de atac**: 5
13. **Frecvenţă**: 5
14. **Impact**: 2
15. **Clasificare generală**: *37*

Valorile numerice de la $(11) - (14)$ *sunt alese de Conway ([3]) pe baza următorelor argumente:*

11. **Dificultatea detecţiei**: *Se referă la dificultatea administratorului de reţea de a depista atacul. Ulterior această evaluare poate fi modificată în jos (odată cu dezvoltarea programelor antivirus) sau în sus (dacă apar noi tipuri de atac mai puternice). Se presupune că reţeaua dispune de cele mai bune tehnici de apărare, iar atacatorul are un nivel mediu de competenţă. Evaluarea* $'4'$ *dată în exemplu se datorează faptului că în acest moment cele mai moderne scanere sunt foarte lente, aşa că majoritatea sistemelor IDS le ignoră.*
12. **Facilitate de atac**: *Estimează dificultatea lansării atacului. Când componentele utilizate în atac sunt free Internet, ratingul creşte. Pentru atacuri unde toate ingredientele sunt securizate (nu sunt free public) – cazul viermilor – valoarea acestui indice scade. Ratingul din exemplu subliniază faptul că un atac "Verif/scan" este uşor de lansat.*
13. **Frecvenţă**: *Arată cât de des întâlnit este atacul în zona reţelelor unde poate produce efectiv pagube. De exemplu, un atac ARP (Address Resolution Protocol) de redirecţionare poate avea o rată de frecvenţă medie, chiar dacă nu este lansat aproape niciodată în conexiuni Internet (nu poate traversa routerele). Atacul din exemplu apare extrem de frecvent (aproape zilnic) în toate reţelele; deci rata sa este* $'5'$.
14. **Impact**: *Evaluează pagubele potenţiale ale unui atac reuşit.Valoarea depinde evident de importanţa modulelor din sistem afectate. Dacă există un atac foarte periculos facilitat de atacul analizat, ratingul său poate creşte, chiar dacă atacul este aproape inofensiv. În cazul de faţă, "Verif/Scan" ar trebui să aibă un impact* $'1'$*, dar pentru că el permite succesul altor atacuri mult mai periculoase, ratingul a fost crescut la* $'2'$.
15. **Clasificare generală**: *Este folosit pentru o clasificare comparativă a diferitelor atacuri, listată la sfârşitul capiltolului.*

10.5 Citire

O descriere grafică a clasei de atac "**Citire" (Read)** este (vom marca o clasă cu un cerc, o subclasă cu o elipsă, iar un atac efectiv cu un dreptunghi):



Atacurile din această clasă sunt atacuri primare care au ca scop obţinerea de informaţii despre potenţiala victimă. Ele merg de la aflarea modului de organizare al adreselor IP la scanarea porturilor şi a vulnerabilităţilor lor, fiind urmate de accesarea (prin slăbiciunile detectate) a datelor.

10.5.1 Recunoaştere (Recon)

Sunt primele atacuri desemnate de atacator pentru a afla mai multe informaţii. Ele tatonează securitatea sistemului folosind metode active sau pasive. În aproape toate cazurile informaţiile obţinute sunt exploatate iterativ, iar după ce se obţin suficiente date, sunt urmate de alte atacuri cu potenţial crescut de virulenţă.

1. Baleierea datelor (Data Scavenging)

Este un atac ale cărui caracteristici sunt sumarizate astfel:

1. **Membru al clasei/subclasei**: Citire/Recunoaştere.
2. **Exemplu de implementări**: Utilităţi oferite de reţea: *Whois, Nslookup, Finger, Traceroute, Ping.* (*http://google.com*).
3. **Preliminarii**: Niciunul.
4. **Vulnerabilităţi**: Niciuna.
5. **Utilizare tipică**: Citeşte domeniile IP, serverele DNS, serverele de mail, sistemele publice, punctele de contact etc.
6. **Rezultate atac**: Dezvăluie informaţii.
7. **Posibile atacuri ulterioare**: Verificare şi scanare.
8. *OSI* **Layers**: $3 - 7$
9. **Detectare**: Aproape imposibilă.
10. **Protecţie**: Niciuna
11. **Dificultatea detecţiei**: 5
12. **Facilitate de atac**: 4
13. **Frecvenţă**: 5
14. **Impact**: 1
15. **Clasificare generală**: 32

Atacul de baleiere a datelor este în general primul pas în orice atac deliberat asupra unei reţele. Atacatorul combină utilităţi ale reţelei cu motoare de căutare pe Internet obţinând cât mai multe informaţii primare despre victimă.

Este aproape imposibil de detectat din două motive:

1. Dacă foloseşte utilităţi de reţea (*Ping, Traceroute* etc), volumul de trafic este atât de scăzut încât este imposibil de identificat vreo intenţie de atac; este imposibil de făcut diferenţa între utilizarea legitimă a protocoalelor şi ntenţia de atac.

2. Informaţia obţinută cu *Whois, Nslookup* sau motoarele de căutare Internet este de obicei publică.

Mai mult, adesea informaţia obţinută vine de la servere diferite de cel al victimei (cazul căutărilor *Whois*).

După un atac de baleiere a datelor, atacatorul va putea dispune de

- Adresele IP ale sistemelor critice (WWW, DNS, mail);
- Domeniile IP asignate victimei;
- Providerul Internet (ISP) al victimei.

2. Verificare şi scanare

Caracteristici generale ale atacului "**Verif/Scan**":

1. **Membru al clasei/subclasei**: Citire/Recunoaştere.
2. **Exemplu de implementări**: Nmap (*http://www.insecure.org/nmap*), Nessus (*http://www.nessu*
3. **Preliminarii**: Baleiere date.
4. **Vulnerabilităţi**: Niciuna.
5. **Utilizare tipică**: Citeşte IPurile şi aplicaţiile accesibile din reţeaua victomei.
6. **Rezultate atac**: Dezvăluie informaţii.
7. **Posibile atacuri ulterioare**: Aproape toate.
8. OSI **Layers**: $3 - 7$
9. **Detectare**: IDS şi firewall-uri (cu *log analysis*)..
10. **Protecţie**: Niciuna (un firewall poate limita unele ţinte de scanare, dar nu o scanare în curs pentru un serviciu free la o adresă IP accesibilă).
11. **Dificultatea detecţiei**: 4
12. **Facilitate de atac**: 5
13. **Frecvenţă**: 5
14. **Impact**: 2
15. **Clasificare generală**: 37

Acest atac este adesea numit şi "*Scanarea porturilor*" sau "*Scanarea vulnerabilităţilor*". Atacatorul foloseşte informaţia obţinută prin atacul de baleiere pentru a afla mai multe informaţii despre reţeaua victimă. De obicei se face o scanare a porturilor, urmată de o scanare a vulnerabilităţilor. Folosind un tool ca *Nmap*, atacatorul poate afla:

- Toate adresele IP accesibile ale reţelei victimă;
- O şansă non-neglijabilă de a ghici care OS al fiecărui sistem accesibil este activ;
- Dacă reţeaua este protejată de un firewall şi – dacă da – de ce tip.

Exemplul 10.2. *([3]) Un exemplu de scanare Ping pentru a afla hosturile active:*

```
Starting nmap V. 3.00 (www.insecure.org/nmap/)
Host  (10.1.1.0) seems to be a subnet broadcast address (returned 3 extrapings).
Note -- the actual IP also responded.
Host (10.1.1.1) appears to be up.
Host (10.1.1.12) appears to be up.
Host (10.1.1.22) appears to be up.
Host (10.1.1.23) appears to be up.
Host (10.1.1.101) appears to be up.
Host (10.1.1.255) appears to be a subnet broadcast address (returned 3 extra pings).
Note -- the actual IP also responded.
Nmap run completed -- 256 IP adresses (7 hosts up) scanned in 4 seconds.
```

Deci, în numai 4 *secunde, Nmap a aflat cinci hosturi active în reţea.*

O descriere mai detaliată despre modul în care lucrează *Nmap* se află la [4].

3. War Dialing şi War Driving

Caracteristici generale ale celor două atacuri:

1. **Membru al clasei/subclasei**: Citire/Recunoaştere.
2. **Exemplu de implementări**: War dialers: multe variante; uzual este *Tone Loc*;
   War Driving: *Netstumbler (http://www.netstumbler.com/).*
3. **Preliminarii**: Niciuna.
4. **Vulnerabilităţi**: De utilizare sau ale politicii folosite.
5. **Utilizare tipică**: Găseşte modemurile nesigure sau *AP*-urile wireless conectate la reţeaua victimei.
6. **Rezultate atac**: Creşterea gradului de acces.
7. **Posibile atacuri ulterioare**: Sniffer.
8. *OSI* **Layers**: $1 - 2$
9. **Detectare**: Aproape imposibilă.
10. **Protecţie**: Verificări pentru *AP* ([19]), controale regulate folosind tooluri war-driving. Pentru modeme se recomandă un audit periodic.
11. **Dificultatea detecţiei**: 5
12. **Facilitate de atac**: 4
13. **Frecvenţă**: 3
14. **Impact**: 5
15. **Clasificare generală**: 42

Atacurile "*War Dialing*" şi "*War Driving*" permit atacatorilor să intre în reţeaua victimei fără a fi controlaţi (identificaţi). În primul caz atacatorul baleiează prin telefon prefixele numerelor asignate victimei sau zonei acesteia, căutând conexiuni de modem. Din lista astfel obţinută, el poate ghici ce sisteme sunt conectate. Sunând apoi la aceste numere, un atacator poate trece uşor peste o mare parte din măsurile de securitate ale victimei, el fiind uşor confundat cu un utilizator normal.

Atacul *War Driving* operează similar, singura deosebire fiind aceea că atacatorul foloseşte o antenă montată pe maşină şi se roteşte în jurul locaţiei fizice a victimei. Scopul este de a identifica punctele de acces LAN (AP-urile) slab securizate, prin care atacatorul se poate conecta direct la reţeaua victomei (a se vedea şi [19]).

10.5.2 Sniffer

Caracteristici generale:

1. **Membru al clasei/subclasei**: Citire.
2. **Exemplu de implementări**: Ethereal (*http://www.ethereal.com*).
3. **Preliminarii**: Redirecţionare trafic sau MAC flooding.
4. **Vulnerabilităţi**: Niciuna.
5. **Utilizare tipică**: Citeşte traficul de pe reţea pentru care nu are permisiune. Obţine parole.
6. **Rezultate atac**: Dezvăluie informaţii.
7. **Posibile atacuri ulterioare**: Acces direct.
8. OSI **Layers**: $2 - 7$
9. **Detectare**: Antisniff.
10. **Protecţie**: Criptografică.
11. **Dificultatea detecţiei**: 5
12. **Facilitate de atac**: 5
13. **Frecvenţă**: 3
14. **Impact**: 3
15. **Clasificare generală**: 36

Se consideră un atac *sniffer* orice situaţie când un intrus citeşte fără autorizaţie pachete de date de pe reţea, sau când acestea sunt deviate prin sistemul său. Scopul atacurilor sniffer este în primul rând citirea într-un mod inteligent a informaţiei utile atacatorului, cum ar fi:

- Informaţii de autentificare (parole);
- Patternuri tipice de utilizare din reţeaua victimei;
- Informaţii de gestiune a reţelei;

- Tranzacţii confidenţiale.

Un virus sniffer (cum este *Ethereal*) poate decodifica informaţii până la nivele de aplicaţii foarte joase (protocoale *Gateway* sau TLS/SSL).

Un Sniffing nu este atac propriu zis, el având două puncte slabe:

1. Informaţia surprinsă trebuie să fie text clar. O criptare a datelor constituie un obstacol în citirea lor de către intruşi.

2. Pachetele citite trebuie trimise – sub o formă oarecare – atacatorului. Dacă acesta este local şi într-un mediu partajat (hub Ethernet, wireless), este suficient ca el să-şi plaseze cardul de interfaţă la reţea (NIC). Dacă mediul este protejat, atacatorul trebuie să realizeze în prealabil o deviere a traficului sau un gen de flooding Media Access Control (MAC).

Din acest motiv, unele firewall-uri nu consideră *Sniffer* drept virus, ci numai un instrument folosit de administratorii de reţea pentru a diagnostica a gamă largă de probleme de conexiune.

10.5.3 Acces direct

Caracteristici generale:

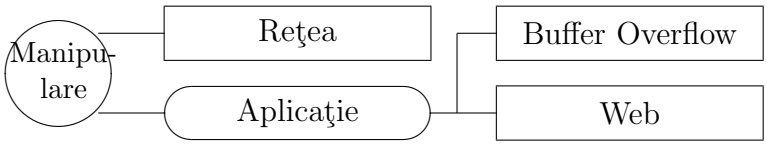
1. **Membru al clasei/subclasei**: Citire.
2. **Exemplu de implementări**: Logare la un server şi furtul fişierului */etc/passwd.*.
3. **Preliminarii**: Diverse.
4. **Vulnerabilităţi**: Niciuna.
5. **Utilizare tipică**: Acces neautorizat la setări de informaţii; furt de date.
6. **Rezultate atac**: Dezvăluie informaţii.
7. **Posibile atacuri ulterioare**: Manipulare.
8. *OSI* **Layers**: 7
9. **Detectare**: IDS.
10. **Protecţie**: Firewall şi aplicaţii de securitate.
11. **Dificultatea detecţiei**: 2
12. **Facilitate de atac**: 5
13. **Frecvenţă**: 5
14. **Impact**: 3
15. **Clasificare generală**: 39

În "*Acces direct*" sunt grupate toate atacurile în care intrusul încearcă să obţină un acces direct la resursele din reţea. De exemplu, dacă a găsit o cale prin care să treacă prin firewall, el va utiliza un atac *Acces direct* pentru a se loga la sistemele protejate de acesta, după care poate lansa o gamă largă de alte atacuri spre alte reţele, cel mai comun fiind un atac de prelucrare a informaţiei.

Deşi *Acces Direct* este aproape întotdeauna lansat pe Nivelul 7, pentru unele aplicaţii el poate fi oprit şi la nivele mai joase.

10.6 Manipulare

Structura clasei de atac "**Manipulare**" este



Această clasă conţine toate atacurile a căror reuşită duce la modificări sau transfer de date – indiferent de nivelul sistemului atacat. Aici se află zeci de atacuri; cea mai mare parte din ele sunt cuprinse în două categorii: atacuri pe reţea şi atacuri pe aplicaţie.

Intruşii folosesc aceste atacuri cu două scopuri:

- Manipulează pachetele de date extrăgând informaţie pe care o prelucrează în alte scopuri;
- Manipulează pachetele de date ale reţelei pentru a-i cauza diverse pagube (maliţioase sau iremediabile).

10.6.1 Atacuri pe reţea

Sunt numeroase atacuri de manipulare a reţelelor. Toate au însă aceleaşi caracteristici (şi de aceea le vom considera ca un singur tip de atac şi nu o subclasă):

1. **Membru al clasei/subclasei**: Manipulare.
2. **Exemplu de implementări**: *Fragroute*.
3. **Preliminarii**: Diverse.
4. **Vulnerabilităţi**: Software.
5. **Utilizare tipică**: Tehnologii de evitare a securităţii.
6. **Rezultate atac**: Creşterea nivelului de acces.
7. **Posibile atacuri ulterioare**: Citire/Composite.
8. *OSI* **Layers**: $3 - 4$.
9. **Detectare**: IDS, routere.
10. **Protecţie**: Firewall, aplicaţii de securitate, criptografie.
11. **Dificultatea detecţiei**: 2
12. **Facilitate de atac**: 3
13. **Frecvenţă**: 2
14. **Impact**: 3
15. **Clasificare generală**: 26

Cel mai frecvent atac de manipulare a reţelei este fragmentarea IP-ului. Aici intrusul fragmentează intenţionat traficul încercând să ocolească (bypass) controlul de securitate – care poate fi al reţelei (IDS sau firewall) sau al unei aplicaţii. Programul tipic care lansează un atac de fragmentare este *Fragroute* ([15]). Detalii asupra diverselor moduri prin care fragmentarea poate fi utilizată la ocolirea barajelor de securitate se găsesc în [7].

Înafara unui atac de fragmentare IP, atacatorul poate executa un *atac la rădăcină (source route attack*). Acesta permite intrusului să transfere drumul atacului prin reţea. Astfel de atacuri nu mai sunt primejdioase, ele fiind respinse automat de majoritatea routerelor.

Protocoalele IP, TCP (Transmission Control Protocol) şi UDP (User Datagram Protocol) sunt deosebit de complexe. Aproape inevitabil, ele lasă posibilitatea atacatorilor să construiască softuri care obligă aceste protocoale să efectueze acţiuni neprevăzute de autorii lor. Dan Kaminsky construieşte un pachet de tooluri numit *Paketto* ([8]) în care arată diverse acţiuni originale care pot fi realizate de TCP/IP.

10.6.2 Atacuri pe aplicaţie

Atacurile de manipulare a aplicaţiei sunt atacuri îndeptate asupra softurilor pe nivele, având ca ţintă exploatarea diverselor slăbiciuni în construirea sau implementarea aplicaţiilor. Cel mai cunoscut este atacul *buffer overflow*. Ulterior s-au dezvoltat şi atacurile aplicaţiilor web pentru reţele – de exemplu *cross-site scripting* sau *insecure Common Gateway Interface* (CGI). Deoarece numărul şi varietatea atacurilor pe aplicaţie este enormă, ne vom restrânge momentan la o analiză doar pentru aceste două tipuri de atac.

Buffer Overflow

Caracteristici:

1. **Membru al clasei/subclasei**: Manipulare/ Atacuri pe aplicaţie.
2. **Exemplu de implementări**: Vulnerabilităţi pe aplicaţii; *http://www.cert.org*.
3. **Preliminarii**: Acces direct.
4. **Vulnerabilităţi**: Software.
5. **Utilizare tipică**: Escaladarea privilegiilor pe maşina ţintă.
6. **Rezultate atac**: Creşterea nivelului de acces.
7. **Posibile atacuri ulterioare**: Citire/Composite.
8. *OSI* **Layers**: 7
9. **Detectare**: IDS, aplicaţii de securitate.
10. **Protecţie**: Aplicaţii de securitate.
11. **Dificultatea detecţiei**: 4
12. **Facilitate de atac**: 3
13. **Frecvenţă**: 5

14. **Impact**: 5

15. **Clasificare generală**: 45

Buffer Overflow este cea mai obişnuită formă de vulnerabilitate de aplicaţie. Ea apare când aplicaţia se dezvoltă neputând să-şi limiteze resursele de memorie utilizate.

De exemplu, un program aşteaptă la intrare 20 octeţi şi primeşte de fapt 300 octeţi. În mod normal, el ar trebui să ignore 280 din ei. Dacă însă aplicaţia are o eroare de cod, ea îi poate suprascrie în altă zonă de memorie şi apoi să execute codul de acolo cu privilegiile aplicaţiei originale. Dacă aplicaţia vulnerabilă lucrează ca root, un atac buffer overflow reuşit va obţine pentru intrus privilegii de root. Pentru detalii privind construirea unui atac *buffer overflow* se poate folosi [6].

Din nefericire, atacurile *buffer overflow*, deşi foarte vech, sunt cele mai periculoase, în principal din cauza pagubelor pe care le provoacă precum şi datorită marilor dificultăţi de a le preveni. Un atac *buffer overflow* lasupra unui site web este lansat sub control de către intrus; deoarece traficul prin portul 80 este permis de firewall, atacatorul va acţiona prin acesta, neputând fi recunoscut decât prin urmările pe care le are.

Web

Carcteristici:

1. **Membru al clasei/subclasei**: Manipulare/ Atacuri pe aplicaţie.

2. **Exemplu de implementări**: *Cross-site scripting*, Aplicaţii *CGI* nesigure.

3. **Preliminarii**: Acces direct.

4. **Vulnerabilităţi**: Software.

5. **Utilizare tipică**: Variabilă.

6. **Rezultate atac**: Creşterea nivelului de acces şi dezvăluirea informaţiei protejate.

7. **Posibile atacuri ulterioare**: Citire/Composite.

8. *OSI* **Layers**: 7

9. **Detectare**: IDS, aplicaţii de securitate.

10. **Protecţie**: Aplicaţii de securitate.

11. **Dificultatea detecţiei**: 3

12. **Facilitate de atac**: 3

13. **Frecvenţă**: 4

14. **Impact**: 3

15. **Clasificare generală**: 33

Atacurile pe aplicaţii Web sunt foarte diverse, cele două menţionate în tabel fiind cele mai cunoscute atacuri de acest gen.

În *Cross-site scripting*, informaţia maliţioasă este ascunsă într-un URL pe care victima va da click. Atacul poate afecta utilizatorii interni ai reţelei care folosesc această adresă. Caracteristic acestui atac web este faptul că nu există o modalitate unitară de a determina locul unde este problema; codul maliţios poate fi pe browserul clientului, pe server, pe site-ul vizitat sau oriunde pe drumul spre acesta. În general, cel mai bun mijloc de prevenire a lui este educarea utilizatorilor de a nu deschide site-uri decât dacă ştiu exact ce conţin acestea. Informaţii detaliate pot fi găsite în zona FAQ de la [10].

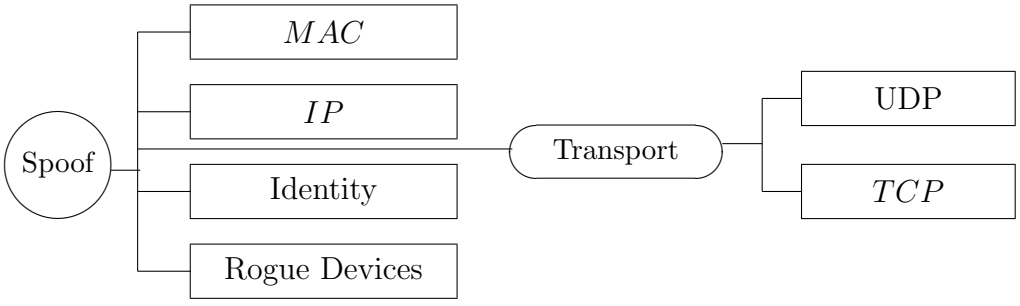
Vom relua mai târziu un atac de acest tip: *Phishing*.

Similar clasei de atacuri *Citire*, aplicaţiile *CGI nesigure* pot constitui primul pas pentru un intrus care caută să compromită un server web. Ori de câte ori se completează un formular sau se inserează adresa proprie pe un website există şanse de a fi victima unei forme de atac CGI. Scripturile CGI curate nu acceptă – printre altele – decât tipuri de date de o formă specificată. De exemplu, dacă programul CGI solicită adresa unui utilizator, el trebuie să-i permită acestuia să folosească numai caractere alfanumerice, punctul şi virgula. Pe de altă parte programul trebuie să permită şi operarea cu caractere cum ar fi $\%, \$$ etc pentru alte scopuri.

Practic, nu există însă programe CGI total sigure, singurul deziderat al dezvoltatorilor de soft fiind acela de a limita numărul de breşe de securitate din aplicaţiile construite. Detalii despre practici de programare cu aplicaţii CGI se pot găsi în [9].

10.7 Spoof

Atacurile *Spoofing* apar atunci când un atacator este capabil să convingă un utilizator/sistem că o anumită informaţie provine de la o sursă care nu a creat-o. O clasificare a lor poate fi sugerată de diagrama:



Un atac spoofing poate fi lansat oriunde există protocoale slabe de autentificare.

10.7.1 MAC Spoofing

Caracteristici:

1. **Membru al clasei/subclasei**: Spoof.
2. **Exemplu de implementări**: Suporturi de reţea care suportă modificarea adresei MAC stocată pe cardul de acces.
3. **Preliminarii**: Acces direct (conectivitate LAN locală).
4. **Vulnerabilităţi**: Niciuna.

5. **Utilizare tipică**: Află o adresă MAC a unui sistem de încredere pentru a trimite sau primi date în numele acestuia.
6. **Rezultate atac**: Creşterea nivelului de acces şi dezvăluirea informaţiei protejate.
7. **Posibile atacuri ulterioare**: Variabilă.
8. OSI **Layers**: 2
9. **Detectare**: Niciuna.
10. **Protecţie**: Memorie de adrese (CAM – Content Addressable Memory) statică.
11. **Dificultatea detecţiei**: 3
12. **Facilitate de atac**: 5
13. **Frecvenţă**: 1
14. **Impact**: 3
15. **Clasificare generală**: 28

MAC Spoofing (altă formă a atacurilor *Layer 2 Spoofing*) este un atac simplu care pur şi simplu pune propria sa adresă MAC în locul aceleia a sistemului considerat de încredere. În mediile Erthernet actuale, tabela CAM de switch-uri păstrează istoricul adreselor MAC, al $VLAN$-urilor şi la ce port este conectată fiecare adresă MAC. Când un intrus schimbă o adresă MAC cu adresa altui sistem deja conectat la switch, tabela CAM se actualizează. Acest lucru apare clar imediat ce atacatorul sistemului trimite un frame pe canal. Tot traficul destinat acestei adrese MAC (şi adresa IP deservită de ea) este returnată atacatorului până când sistemul real comunică din nou.

Atacul lucrează bine pe sisteme care doar primesc date (de exemplu servere *Syslog*). Nu este un atac deosebit de periculor şi stoparea lui este rezonabilă numai în cazul sistemelor critice, unde o linie CAM poate fi configurată astfel ca o adresă MAC dată este întotdeauna asociată unui anumit port.

10.7.2 IP Spoofing

Caracteristici:

1. **Membru al clasei/subclasei**: Spoof.
2. **Exemplu de implementări**: Orice atac capabil să acceseze sistemul de control al driverelor unui sistem.
3. **Preliminarii**: Niciunul.
4. **Vulnerabilităţi**: Niciuna.
5. **Utilizare tipică**: Ascunde sursa un atac de nivel superior.
6. **Rezultate atac**: Creşterea nivelului de acces.
7. **Posibile atacuri ulterioare**: Variabilă.
8. OSI **Layers**: 3

9. **Detectare**: IDS.
10. **Protecţie**: RFC 2827, RFC 1918, verificare RPF (pe router sau firewall), criptografie.
11. **Dificultatea detecţiei**: 3
12. **Facilitate de atac**: 4
13. **Frecvenţă**: 5
14. **Impact**: 1
15. **Clasificare generală**: 30

După cum se ştie, un header IP are 20 octeţi (Capitolul ?? - $TCP - IP$). Niciunul din câmpurile sale nu este protejat pentru un atac spoofing. Dacă intrusul poate accesa sistemul de control al driverelor (care de obicei este admis doar pentru administrator sau pentru *root*), el poate trimite un pachet cu orice header IP. Există pe Internet numeroase aplicaţii şi biblioteci puse free la dispoziţie atacatorilor sau administratorilor de sistem care doresc să testeze securitatea. Cele mai cunoscute sunt:

- **Libnet**: $http://www.packetfactory.net/libnet/$;
- **Hping**: $http://www.hping.org/$.

Ca şi atacul Spoofing precedent, *IP Spoofing* este cauzat în principal de unele vulnerabilităţi software. Factorul de impact este relativ redus dacă atacul este izolat. Totuşi, dacă el este o componentă a unui atac mai complicat, poate deveni extrem de periculos.

Criptografia este folosită ca mecanism de protecţie numai atunci când se aplică unui sistem care necesită comunicaţii criptate pentru accesul la informaţiile IP. De exemplu, într-o aplicaţie financiară, cumpărătorul trebuie să afle IP-ul unui vânzător fără a-şi dezvălui identitatea.

10.7.3 Transport Spoofing

Această subclasă de atacuri se referă la reuşita unui spoofing pe nivelul de transport (Layer 4). Sunt cuprinse aici două mari tipuri de atac: UDP **Spoofing** şi TCP **Spoofing**.

UDP Spoofing

Caracteristici:

1. **Membru al clasei/subclasei**: Spoof/Transport Spoofing.
2. **Exemplu de implementări**: Orice atac capabil să acceseze sistemul de control al driverelor unui sistem.
3. **Preliminarii**: IP Spoofing.
4. **Vulnerabilităţi**: Niciuna.
5. **Utilizare tipică**: Introduce date neautorizate într-o aplicaţie care utilizează UDP ca mijloc de transport.
6. **Rezultate atac**: Coruperea sau dezvăluirea informaţiei.

7. **Posibile atacuri ulterioare**: Variabilă.
8. *OSI* **Layers**: 4
9. **Detectare**: Niciuna (Trebuie stopat IP Spoofing).
10. **Protecţie**: Se folosesc proprietăţi din TCP sau se blochează atacul IP Spoofing.
11. **Dificultatea detecţiei**: 5
12. **Facilitate de atac**: 4
13. **Frecvenţă**: 3
14. **Impact**: 3
15. **Clasificare generală**: 34

Un Header UDP este mult mai simplu decât un header IP. El conţine numai 8 octeţi împărţiţi în 4 câmpuri: numerele celor două porturi (sursă şi destinaţie), lungimea câmpului de date, şi o sumă de control (opţională la unele sisteme mai vechi). Nu există nici o noţiune de conectivitate. De aceea un UDP este mai simplu de atacat. Cum managementul multor aplicaţii cum ar fi $SNMP$ (Network Management Protocol), *Syslog*, $TFTP$ (Trivial File Transfer Protocol) foloseşte UDP ca mecanism de transport, securitatea administrării canalelor unei reţele este considerată ca fiind veriga slabă a securităţii wireless.

TCP Spoofing

Caracteristici:

1. **Membru al clasei/subclasei**: Spoof/Transport Spoofing.
2. **Exemplu de implementări**: Orice atac capabil să acceseze sistemul de control al driverelor unui sistem.
3. **Preliminarii**: IP Spoofing.
4. **Vulnerabilităţi**: Niciuna.
5. **Utilizare tipică**: Introduce date neautorizate într-o aplicaţie care utilizează TCP ca mijloc de transport.
6. **Rezultate atac**: Coruperea sau dezvăluirea informaţiei.
7. **Posibile atacuri ulterioare**: Variabilă.
8. *OSI* **Layers**: 4
9. **Detectare**: Niciuna (Trebuie stopat IP Spoofing).
10. **Protecţie**: Se blochează atacul IP Spoofing.
11. **Dificultatea detecţiei**: 5
12. **Facilitate de atac**: 1
13. **Frecvenţă**: 1

14. **Impact**: 5
15. **Clasificare generală**: 30

Din structura unui header TCP (Capitolul ? - $TCP - IP$) se observă că acesta este mult mai complex decât headerul UDP. În particular, el conţine un nonce – un număr aleator pe 32 biţi ataşat conexiunii. Fără un acces direct la şirul de date printr-un atac *sniffing*, este imposibil de ghicit valoarea lui. Pentru a insera o comunicare validă în fluxul de date, atacatorul trebuie să ştie această valoare şi – simultan – să întrerupă legătura dintre client şi server.

Dacă intrusul este exterior, atacul are foarte puţine şanse de reuşită. El devine însă mult mai periculos dacă este lansat din interior, dintr-o locaţie aflată pe drumul dintre client şi server. Fiind conectat la server, atacatorul îşi poate crea acces la toată informaţia necesară pentru lansarea atacului.

10.7.4 Identity Spoofing

Caracteristici:

1. **Membru al clasei/subclasei**: Spoof.
2. **Exemplu de implementări**: $LC4$ (*http://www.atstake.com*); Jack Spintecătorul (*http://www.openwall.com/john/*).
3. **Preliminarii**: IP Variabile.
4. **Vulnerabilităţi**: Utilizare.
5. **Utilizare tipică**: Convinge o reţea că este utilizator legal.
6. **Rezultate atac**: Creşterea nivelului de acces.
7. **Posibile atacuri ulterioare**: Citire/Manipulare.
8. OSI **Layers**: 7
9. **Detectare**: Niciuna.
10. **Protecţie**: Aplicaţii de securitate (pentru detectare şi protecţie).
11. **Dificultatea detecţiei**: 4
12. **Facilitate de atac**: 3
13. **Frecvenţă**: 4
14. **Impact**: 5
15. **Clasificare generală**: 42

Prin *Identity Spoofing* se înţeleg diverse tipuri de atac, cum ar fi spargerea de parole, încercări de logare brute-force, furt de certificate digitale etc. Evaluările numerice din tabelă se referă la cea mai frecventă formă de furt de identitate de pe reţea: cea de username şi de parolă. Ţinta lor sunt mecanismele de identificare, aflate pe un nivel superior. Dacă intrusul reuşeşte să le compromită, el poate coborî uşor – prin intermediul mai multor biţi de informaţie partajaţi – pe nivele mai joase, de aplicaţii.

O listă a mecanismelor de identificare este (în ordinea crescătoare a securităţii):

1. Username şi parolă în clar (de exemplu *Telnet*);
2. Chei prestabilite (de exemplu WEP – Wired Equivalent Privacy);
3. Parole One-Time;
4. Criptografia cu chei publice (de exemplu $PGP, IPSec$).

Observaţia 10.1. *Lista se referă numai la mecanismele de identificare, nu la puterea sistemului care le gestionează. De asemenea, "text clar" şi "text criptat" se referă la parolele aflate în tranzit, nu la cele stocate pe serverul sistemului. Astfel, parolele folosite de Telnet pe sistemul Unix sunt stocate sub formă criptată chiar dacă prin reţea ele trec sub formă de text clar.*

Jack Spintecătorul şi $LC4$ sunt atacuri de spargere de parole, care – în linii mari – încearcă să ghicească o parolă, apoi o criptează şi o compară cu versiunea criptată a parolei victimei stocată pe server. Majoritatea parolelor sunt stocate sub formă de amprentă printr-o funcţie de dispersie criptografică. Pe baza lor, cel mai simplu mod de a afla o parolă este de a aplica funcţia de dispersie unor parole succesive folosind un atac numit "*dictionary attack*". Detaliind:

1. Se obţine o listă $\{h(x_1), h(x_2), \ldots, h(x_n)\}$ de parole.
   De obicei aceasta este cea mai dificilă etapă, deoarece atacatorul trebuie să obţină acces la root pentru a copia din lista de parole amprentate. Pe Unix de exemplu, ea se află stocată în $/etc/shadow$.
2. Se ia o amprentă $h(x_i)$.
   De exemplu $h(x_i) = D3bgT57Pff2BgSK56wQI8Y0X1Vkp$.
3. Se generează diverse parole $a_1, a_2, \ldots$ şi se calculează $h(a_1), h(a_2), \ldots$ până se găseşte un j astfel ca $h(a_j) = h(x_i)$, din care intrusul determină parola a_j. Atacul bazat pe paradoxul naşterilor este foarte util aici.

Identity spoofing este unul din cele mai periculoase atacuri, în primul rând deoarece administratorul de sistem nu poate obliga utilizatorii să aleagă parole dificile. Cât timp un utilizator trebuie să ţină minte o parolă, el nu va alege una dificilă, ci una legată intrisec de cunoştinţele sale. O soluţie alternativă ar fi identificările biometrice (cu alte slăbiciuni însă).

10.7.5 Aparate false (Rogue Devices)

Caracteristici:

1. **Membru al clasei/subclasei**: Spoof.
2. **Exemplu de implementări**: Orice device legitim de reţea.
   De exemplu $WLAN\ AP, DHCP$, server, router, host.
3. **Preliminarii**: Acces fizic la device.
4. **Vulnerabilităţi**: Utilizare sau control fizic al securităţii.
5. **Utilizare tipică**: Oferă servicii comunităţii; datele furate din cererile lor sunt trecute apoi spre reţeaua legitimă.
6. **Rezultate atac**: Dezvăluirea şi coruperea informaţiei.

7. **Posibile atacuri ulterioare**: Citire/Manipulare.
8. *OSI* **Layers**: Toate.
9. **Detectare**: Variază după tipul device-ului.
10. **Protecţie**: Variază după tipul device-ului.
11. **Dificultatea detecţiei**: 3
12. **Facilitate de atac**: 2
13. **Frecvenţă**: 2
14. **Impact**: 5
15. **Clasificare generală**: 33

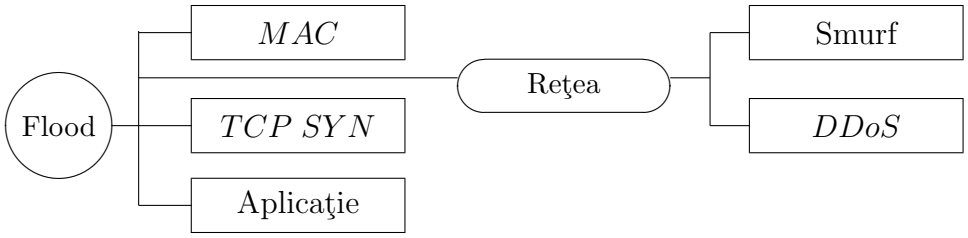
Spre deosebire de atacurile spoofing precedente – bazate exclusiv pe slăbiciuni software – acest atac introduce în reţea un device propriu (fals), sperând să convingă utilizatorii şi device-urile din reţea că este legitim. Cele mai comune astfel de atacuri sunt bancomatele false şi cardurile de acces (bancare, telefonice etc).

Proiectul *DC Phone* ([11]) descrie detaliat modalitatea prin care un intrus poate introduce un $PC, SegaDreamcast$ sau un *Compaq iPAQ* într-o reţea. Aici el încearcă să determine adresa IP şi prezenţa unui server proxy $HTTP$, după care crează o conexiune tunnelată spre intrus. În continuare sunt posibile alte diverse atacuri, cum ar fi *Redirecţionare ARP* sau MAC *Flooding*.

Un atac cu aparat fals poate fi extrem de periculos, dar montarea aparatului este dificilă, deoarece intrusul trebuie să aibă acces fizic la reţeaua pe doreşte să o compromită.

10.8 Flood

Un atac de tip "*Flood*" apare atunci când un atacator trimite un număr foarte mare de date care blochează resursele reţelei. O taxonomie a lor este reprezentată prin graful:



Sunt atacuri diverse, extrem de frecvente, cu grade diferite de periculozitate. Scopul lor nu este aflarea sau coruperea informaţiei, ci blocarea ei şi a proceselor reţelei atacate.

10.8.1 MAC Flooding

Caracteristici:

1. **Membru al clasei/subclasei**: Flood.
2. **Exemplu de implementări**: $macof$.
3. **Preliminarii**: Acces LAN local.

4. **Vulnerabilităţi**: Politica de securitate.
5. **Utilizare tipică**: Umple o tabelă CAM şi apoi respinge traficul legitim.
6. **Rezultate atac**: Dezvăluirea informaţiei.
7. **Posibile atacuri ulterioare**: Citire/Manipulare.
8. OSI **Layers**: 2
9. **Detectare**: Schimbarea monitorizării (mărimea tabelei CAM).
10. **Protecţie**: Securizarea portului.
11. **Dificultatea detecţiei**: 3
12. **Facilitate de atac**: 5
13. **Frecvenţă**: 1
14. **Impact**: 3
15. **Clasificare generală**: 28

Un atac "MAC *flooding*" trimite pachete cu sursă ascunsă şi destinaţie adresele MAC, de la sistemul atacatorului spre reţeaua Ethernet. Tabela CAM care păstrează urma adreselor MAC are mărime limitată. După ce ea s-a umplut, frame-urile destinate adreselor MAC dar nu au o intrare în CAM sunt revărsate în reţeaua $VLAN$ locală pentru a fi livrate destinatarului. În acest fel pachetele de informaţii sunt prelucrate similar unor informaţii partajate şi pot fi accesate de intrus.

10.8.2 Umplere reţea

Atacurile de umplere a reţelei (*Network flooding*) au ca scop consumarea benzii de transmisie a unei reţele. După aceea, şansele ca traficul legitim să circule între utilizatori va fi extrem de redus. Atacurile sunt îndreptate în special asupra conexiunilor reţelelor Internet, caracterizate deja prin viteză mică de procesare. Vom analiza două astfel de atacuri (cele mai frecvente).

Smurf

Caracteristici:

1. **Membru al clasei/subclasei**: Flood/Umplere reţea.
2. **Exemplu de implementări**: Aproape orice program *ping*.
3. **Preliminarii**: Acces la reţea, abilitate de a "fura" o adresă a victimei.
4. **Vulnerabilităţi**: Politica de securitate.
5. **Utilizare tipică**: Umple o conexiune a unui site Internet cu $ICMP$ *echo* dat de traficul de răspuns.
6. **Rezultate atac**: $DDoS$.
7. **Posibile atacuri ulterioare**: Niciunul.

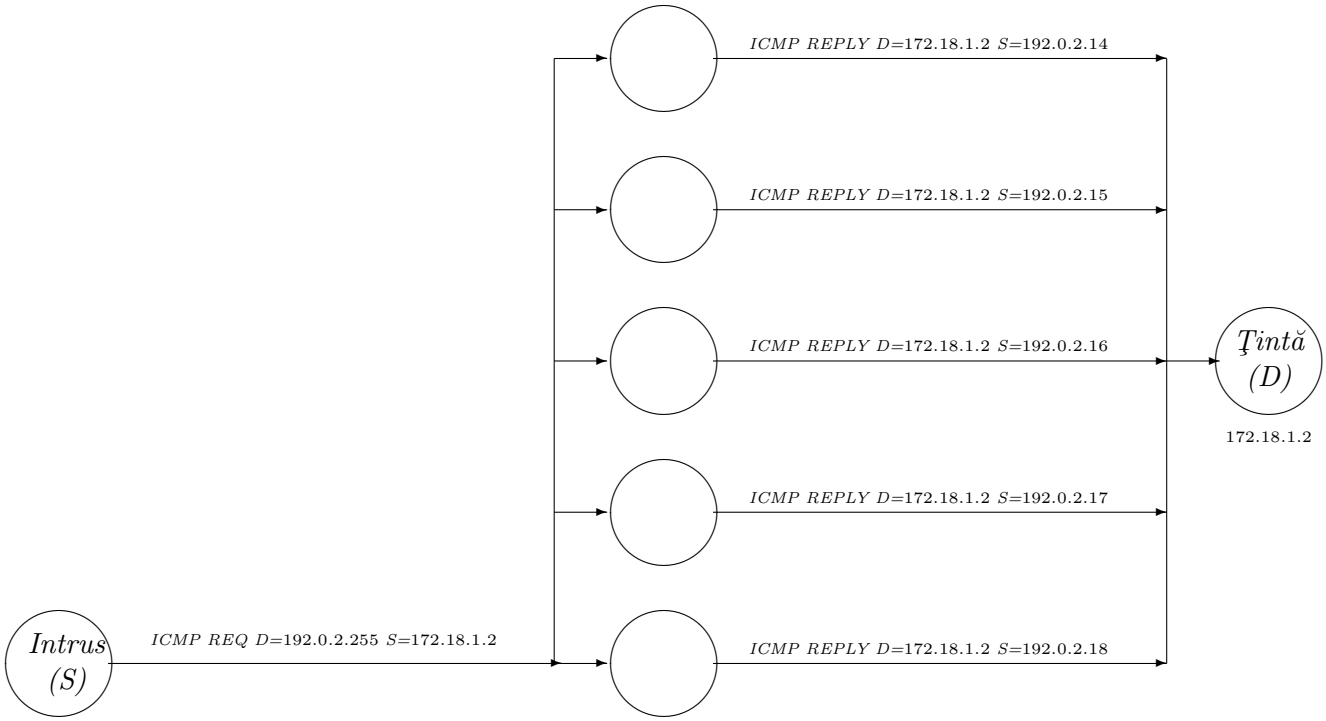
8. *OSI* **Layers**: 3
9. **Detectare**: IDS, log analysis.
10. **Protecţie**: Implementarea comenzii **no ip directed-broadcast**; Alegerea unor stive gazdă pentru IP care să nu răspundă la ping-urile broadcast; CAR (Commited access rate).
11. **Dificultatea detecţiei**: 2
12. **Facilitate de atac**: 4
13. **Frecvenţă**: 2
14. **Impact**: 3
15. **Clasificare generală**: 28

Atacul "*smurf*" (numit după personajele de desene animate) foloseşte ping-urile broadcast mascate prin Internet Control Message Protocol ($ICMP$). Un IP conţine informaţii utile pentru un *direct broadcast*.

Un **direct broadcast** apare când o staţie a unei reţele trimite un pachet broadcast spre altă reţea. De exemplu, o staţie din reţeaua 192.0.2.0/24 trimite un pachet spre 192.0.3.255. Dacă routerul este configurat să propage direct broadcast, reţeaua 192.0.3.0/24 primeşte acest pachet şi-l retrimite tuturor staţiilor pe care le controlează. Acestea vor răspunde toate staţiei sursă.

Atacul *smurf* foloseşte acest comportament pentru a multiplica un pachet mic (*smurf*) de date într-unul mare, capabil să blocheze o staţie.

Exemplul 10.3. *Să considerăm atacul din figură:*



S trimite o cerere de ecou $ICMP$ spre adresa broadcast a unei reţele (aceasta nu este ţinta atacului, ci numai o trambulină de amplificare). Adresa sursă a pachetului $ICMP$ este de la un device al victimei (D). Fiecare staţie din reţea va trimite un pachet ping de

răspuns spre ţintă, amplificând enorm cantitatea de informaţie care trebuie depozitată de aceasta.

De exemplu, dacă S generează un pachet sub forma unui şir de 768 kbs spre o reţea cu 100 staţii, acestea vor trimite spre victimă un şir de 76,8 Mbs.

O comandă **no ip directed-broadcast** va preveni apariţia acestui comportament al reţelei şi o va proteja de atacul smurf.

DDoS

Caracteristici:

1. **Membru al clasei/subclasei**: Flood/Umplere reţea.
2. **Exemplu de implementări**: *Tribe Flood Network 2000, Shaft.*
3. **Preliminarii**: Abilitatea de a infecta un număr mare de sisteme cu care se construieşte o reţea de atacatori zombie.
4. **Vulnerabilităţi**: Politica de securitate.
5. **Utilizare tipică**: Sufocă conexiunea victimei cu o cantitate mare de mesaje Internet.
6. **Rezultate atac**: Respingerea serviciului Internet.
7. **Posibile atacuri ulterioare**: Niciunul
8. OSI **Layers**: $3 - 4$
9. **Detectare**: IDS, log analysis.
10. **Protecţie**: CAR, filtre specifice, opţiuni ISP.
11. **Dificultatea detecţiei**: 2
12. **Facilitate de atac**: 2
13. **Frecvenţă**: 3
14. **Impact**: 4
15. **Clasificare generală**: 31

Din familia de atacuri pe Internet, atacurile $DDoS$ au cea mai mare notorietate, mai ales datorită atacurilor recente asupra site-urilor guvernamentale din diverse ţări (Estonia). Vom detalia mai târziu într-o secţiune separată acest atac, menţionând aici numai forma standard de desfăşurare (reliefată de atacul Stachelraht).

1. Atacatorul infectează un număr de servere de pe Internet şi injectează în ele un virus $DDosS$.
2. Serverele infectate subjugă sistemele aflate sub control transformându-le în agenţi proprii (*boţi*). Atacul folosit poate fi de orice fel, (de exemplu un *troian* prin e-mail) exploatând o vulnerabilitate de aplicaţie sau a sistemului de operare.
3. La un moment, atacatorul trimite ordinul de atac serverelor infectate, cu data atacului şi adresa IP a victimei.
4. Toate sistemele infectate lansează simultan o avalanşă de mesaje de date spre calculatorul vizat, care este sufocat de trafic.

10.8.3 $TCP\ SYN$ Flooding

Caracteristici:

1. **Membru al clasei/subclasei**: Flood.
2. **Exemplu de implementări**: *Apsend, Spastic.*
3. **Preliminarii**: Acces direct.
4. **Vulnerabilităţi**: Software.
5. **Utilizare tipică**: Sufocă o gazdă anumită cu cereri de conectare.
6. **Rezultate atac**: DoS.
7. **Posibile atacuri ulterioare**: de tip Spoof şi Device-uri false.
8. OSI **Layers**: 4
9. **Detectare**: IDS, log analysis, aplicaţii de securitate.
10. **Protecţie**: Cookie $TCP\ SYN$, interceptoare TCP.
11. **Dificultatea detecţiei**: 3
12. **Facilitate de atac**: 5
13. **Frecvenţă**: 3
14. **Impact**: 2
15. **Clasificare generală**: 30

$TCP\ SYN$ este unul din primele forme de atac flooding. Atacatorul trimite un pachet $TCP\ SYN$, după care nu răspunde deloc la pachetul $SYN - ACK$ trimis ca răspuns. Serverul accesat păstrează cererea şi retrimite de câteva ori pachetul $SYN - ACK$ înainte de a închide încercarea (falsă) de stabilire a sesiunii de comunicare.

Când atacatorii lansează un atac $TCP\ SYN$, ei trimit sistemului mii de cereri de conectare. În încercarea acestuia de a răspunde la toate aceste cereri, sistemul îşi consumă toată memoria şi cedează. În primii ani, acest atac era uşor de realizat deoarece mărimile cozilor de conectare ale sistemelor erau mici (primul astfel de atac – celebru – al lui Kevin Mitnick asupra computerelor Tsutomu Shimomura a necesitat numai 8 cereri $TCP\ SYN$ pentru a umple coada unui calculator). Astăzi sistemele rezistă mult mai bine acestor atacuri, datorită atât îmbunătăţirii aplicaţiilor şi sistemelor de operare, cât şi a dezvoltării de tehnologii (cookie-uri $TCP\ SYN$ şi interceptoare TCP).

De remarcat că un atac flooding de acest tip nu este posibil asupra UDP. Deoarece UDP nu are nici o legătură cu operaţia de conectare, un atac UDP flooding va produce pagube minore la nivelul IP. Asupra UDP-ului sunt mult mai puternice atacurile de tip $DDoS$.

10.8.4 Flooding pe aplicaţie

Caracteristici:

1. **Membru al clasei/subclasei**: Flood.
2. **Exemplu de implementări**: Spam, Flooding pe autentificare, Abuz de procesare CPU.
3. **Preliminarii**: Acces direct.
4. **Vulnerabilităţi**: Oricare.
5. **Utilizare tipică**: Face inutilizabilă o aplicaţie sau sistem.
6. **Rezultate atac**: DoS.
7. **Posibile atacuri ulterioare**: de tip Spoof şi Device-uri false.
8. OSI **Layers**: 7
9. **Detectare**: IDS, log analysis, aplicaţii de securitate.
10. **Protecţie**: Aplicaţie de securitate.
11. **Dificultatea detecţiei**: 3
12. **Facilitate de atac**: 5
13. **Frecvenţă**: 5
14. **Impact**: 2
15. **Clasificare generală**: 36

Flooding-urile pe aplicaţie se referă la paleta de atacuri desemnate să scoată din circuit o aplicaţie sau un sistem prin consumarea resurselor sale. Exemplul tipic este *Spamul*. Deşi spamul nu este abilitat pentru consumul de resurse, el are totuşi acest efect asupra sistemului de poştă electronică.

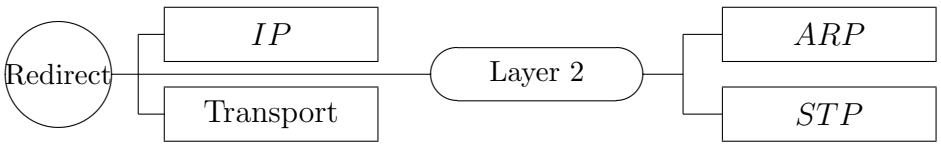
Alt tip de atac se referă la o rulare intensivă continuă a CPU sau un flooding pe server cu cereri de autentificare nefinalizate (atacatorul iniţiază o conexiune, apoi nu mai răspunde din momentul stabilirii parolei); acesta seamănă cu atacul SYN flood, singura diferenţă fiind layerul de aplicaţie.

Alt atac interesant (numit şi *Flash crowds*) este bazat pe efectul Slashdot. Când este dată o ştire interesantă, de obicei sunt descrise doar câteva idei, după care se face trimitere la un link sau o locaţie www unde există mai multă informaţie. Dacă ştirea este populară, vor exista foarte multe conexiuni legitime spre site, blocându-l o vreme.

10.9 Redirect

Într-un atac "*Redirect*" adversarul încearcă schimbarea fluxului de informaţie dintr-o reţea. Aceasta poate apare la orice nivel, dar din perspectiva ideii de securitate pe reţea sunt importante numai redirectarea pentru *Layer* 2, IP şi transport.

O taxonomie a clasei *Redirect* este:



10.9.1 Redirectare *L2*

O redirectare pe Nivelul 2 poate fi realizată folosind *ARP* sau *Spanning Tree Protocol* (*STP*). Le vom trata pe amândouă.

ARP Redirectare/Spoofing

Caracteristici:

1. **Membru al clasei/subclasei**: Redirectare/ Layer 2.
2. **Exemplu de implementări**: *arpsoof (part* sau *dsniff*).
3. **Preliminarii**: Acces direct (conectivitate *LAN* locală),
4. **Vulnerabilităţi**: Niciuna.
5. **Utilizare tipică**: Redirecţionează traficul de ieşire din reţea prin sistemul atacatorului în loc de gateway ales ca default.
6. **Rezultate atac**: Dezvăluirea informaţiei.
7. **Posibile atacuri ulterioare**: Manipulare/Citire.
8. *OSI* **Layers**: 2
9. **Detectare**: *IDS*, *arpwatch*.
10. **Protecţie**: Inspecţii *ARP* şi *ARP* static.
11. **Dificultatea detecţiei**: 3
12. **Facilitate de atac**: 5
13. **Frecvenţă**: 1
14. **Impact**: 4
15. **Clasificare generală**: 30

De obicei acest atac este cunoscut sub numele "*ARP Spoofing*". Totuşi în contextul taxonomiei, prima sa funcţie este redirecţionarea traficului; spoofing-ul este numai un mecanism utilizat pentru atac. Atacul funcţionează astfel: adversarul trimite o avalanşă de broadcast-uri *ARP* afirmând că adresa *MAC* a gateway-ului ales ca default s-a schimbat cu adresa *MAC* a atacaotrului. După ce maşina victimei updatează cache-ul *ARP*, toate cererile de ieşire ale sistemului victimei sunt îndreptate spre sistemul atacatorului, unde apoi mesajele pot fi citite, modificate sau şterse.

Redirectare *STP*

Caracteristici:

1. **Membru al clasei/subclasei**: Redirectare/ Layer 2.
2. **Exemplu de implementări**: Orice device capabil să genereze mesaje *STP* (switch, host UNIX etc)
3. **Preliminarii**: Acces direct (conectivitate *LAN* locală),

4. **Vulnerabilităţi**: Politica de securitate.
5. **Utilizare tipică**: Schimbă drumul prin nivelul 2 al reţelei prin includerea sistemului atacator ca punct de comutare (switch).
6. **Rezultate atac**: Dezvăluirea informaţiei.
7. **Posibile atacuri ulterioare**: Manipulare/Citire.
8. *OSI* **Layers**: 2
9. **Detectare**: Tooluri de management a reţelei.
10. **Protecţie**: Control la *STP* root şi/sau *BTPU*.
11. **Dificultatea detecţiei**: 3
12. **Facilitate de atac**: 3
13. **Frecvenţă**: 1
14. **Impact**: 2
15. **Clasificare generală**: 20

Atacurile *STP* pot fi utilizate ca o altă modalitate de redirecţionare a traficului la Nivelul 2. Atacatorul este capabil să convingă acest nivel al reţelei că el este o rădăcină de bridge *STP*. Efectul este o modificare a topologiei nivelului 2 într-o structură avantajoasă intrusului.

10.9.2 Redirectare *IP*

Caracteristici:

1. **Membru al clasei/subclasei**: Redirectare.
2. **Exemplu de implementări**: Orice device capabil să ruleze protocoale de routare.
3. **Preliminarii**: Acces direct (sistemul de routare al victimei trebuie să fie accesibil).
4. **Vulnerabilităţi**: Configurare/Execuţie.
5. **Utilizare tipică**: Introduce drumuri de routare preferenţiale sau modifică configuraţia de router pentru a transfera traficul prin sistemul atacatorului.
6. **Rezultate atac**: Dezvăluirea informaţiei.
7. **Posibile atacuri ulterioare**: Manipulare/Citire.
8. *OSI* **Layers**: 3
9. **Detectare**: Tooluri de management a reţelei.
10. **Protecţie**: Autentificare router şi sisteme de management pentru router.
11. **Dificultatea detecţiei**: 2
12. **Facilitate de atac**: 2

13. **Frecvenţă**: 2
14. **Impact**: 4
15. **Clasificare generală**: 28

Un atac prin redirectare IP poate schimba fluxul de informaţii în două moduri:

- Introducând un router fals cu advertizări nelegitime;
- Reconfigurând routerele existente.

Traficul va trece acum prin sistemul atacatorului care are astfel acces direct la el şi poate citi, modifica sau şterge mesajele.

O securizare atentă a protocolului de routare şi a comenzilor de control pe routere poate reduce drastic rata de succes ale unor astfel de atacuri.

10.9.3 Redirectare transport

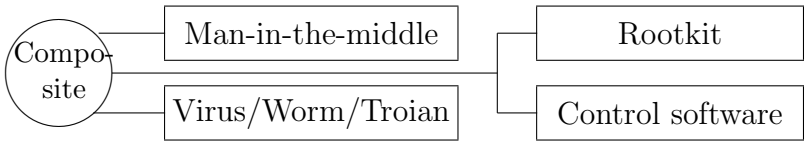
Caracteristici:

1. **Membru al clasei/subclasei**: Redirectare.
2. **Exemplu de implementări**: *Netcat*.
3. **Preliminarii**: Variabile.
4. **Vulnerabilităţi**: Variabile.
5. **Utilizare tipică**: Redirectează cererile la un număr de port şi adresele IP la alt număr de port.
6. **Rezultate atac**: Creşterea accesulu.i
7. **Posibile atacuri ulterioare**: Manipulare/Citire.
8. OSI **Layers**: 4
9. **Detectare**: IDS
10. **Protecţie**: Aplicaţii de securitate.
11. **Dificultatea detecţiei**: 4
12. **Facilitate de atac**: 3
13. **Frecvenţă**: 2
14. **Impact**: 3
15. **Clasificare generală**: 28

"*Redirectarea transportului*" – numit şi "*Redirectarea portului*"– este un atac prin care intrusul poate intercepta traficul care este deviat pe alt drum. Atacatorul este capabil să seteze în sistemul compromis un soft care va redirecţiona cererile de la un sistem şi port (legitim) la alt sistem şi alt port (ilegitim). Cel mai bun exemplu de funcţionare este *Netcat*, accesibil la [16].

10.10 Composite

Atacurile descrise în această secţiune folosesc o combinaţie din mai multe tipuri de atac. O structură generală a clasei este:



10.10.1 Man-in-the-middle

Atacul *Man-in-the-middle* ($MITM$) este unul din cele mai cunoscute, mai ales din domeniul criptografiei cu chei publice. Pentru reţelele wireless, caracteristicile sale sunt:

1. **Membru al clasei/subclasei**: Composite.
2. **Exemple de implementări**: *dsniff, Ettercap.*
3. **Preliminarii**: Variabile.
4. **Vulnerabilităţi**: Variabile.
5. **Utilizare tipică**: Urmăreşte traficul şi atacă sesiunile.
6. **Rezultate atac**: Variabilă (orice este posibil).
7. **Tipuri de atac folosite**: Citire, Manipulare, Spoof, Redirect.
8. *OSI* **Layers**: 7
9. **Detectare**: Variabilă.
10. **Protecţie**: Criptografie.
11. **Dificultatea detecţiei**: 4
12. **Facilitate de atac**: 2
13. **Frecvenţă**: 1
14. **Impact**: 5
15. **Clasificare generală**: 31

Observaţia 10.2. *În analiza de sus s-a înlocuit rubrica "Posibile atacuri ulterioare" cu "Tipuri de atac folosite" în care se listează ce atacuri prezentate anterior sunt înglobate aici.*

Într-un atac $MITM$ pe reţele wireless, atacatorul are control activ asupra celui mai înalt nivel relevant de conversaţie dintre două victime. Acesta este de obicei Nivelul 7, dar uneori el poate apare şi la nivelul 3 (când se utilizează $IPSec$).

Exemplul 10.4. *Într-o comunicare între un client şi bancă, clientul crede că este în legătură cu banca, iar banca – cu clientul. În realitate ambele conversaţii sunt routate prin sistemul atacatorului, care poate modifica datele personale, sumele transferate etc.*

Un sistem de criptare bine implementat constituie cel mai simplu mod de apărare contra unui atac $MITM$. Vom da ca exemplu două studii de caz: *dsniff* şi *Ettercap.*

dsniff

Sub acest nume sunt adunate o serie de tool-uri construite de Dug Song (pot fi descărcate de pe [12]). Fiecare element poate fi utilizat pentru a realiza propriul său atac. Astfel, *macof* poate lansa *MAC flooding, arpspoof* poate lansa *ARP Redirection* şi *spoofing*, tool-ul *dsniff* poate activa ca un sniffer selectiv şi afla username şi parola unui utilizator. Lucrând împreună, toolurile din *dsniff* permit intrusului să lanseze un aac $MITM$ complet.

Sunt posibile diverse tipuri de atac $MITM$. De exemplu, cu $arpspoof, dnsspoof$ şi *webmitm* se poate construi următorul atac:

1. Atacatorul începe prin a rula *arpspoof* pentru a redirecţiona prin propria sa maşină traficul îndreptat spre principalul gateway.

2. După ce acest trafic traversează maşina atacatorului, *dnsspoof* returnează adresa IP a atacatorului cu datele DNS specifice hosturilor din spatele sistemului atacator. Deoarece un browser web arată un nume şi nu o adresă IP, victima nu va şti că cererile sale sunt routate prin sistemul intrusului şi nu spre adevărata destinaţie.

3. La sosirea unei cereri web este lansat *webmitm*; acesta generează un certificat digital auto-semnat pe care îl prezintă victimei atunci când la un server SSL vine o cerere de conectare. În acest punct, cererile web sunt trimise proxy preferenţial spre sistemul atacatorului şi nu direct spre un web proxy comercial. Principala diferenţă este aceea că, dacă victima nu observă certificatul fals (mulţi utilizatori fac click pe "Yes" ori de câte ori apare un certificat în browserul lor web), atunci atacatorul este capabil să citească toate pachetele şi eventual să le modifice înainte de a le retrimite serverului real.

4. Atacul este complet. Victima crede că vorbeşte cu compania sa, dar de fapt este conectată la sistemul atacatorului, care s-a intercalat pe traficul victimei.

Ethercap

Este un tool similar cu *dsniff*, care poate fi găsit la [13]. Principalele diferenţe faţă de *dsniff* sunt:

- Abilitatea atacatorului de a face *ARP spoof* la ambele capete ale canalului de comunicare, asigurând trecerea prin propria maşină atât a traficului trimis cât şi a celui primit.

- Inserarea de comenzi în timp real în sesiuni TCP, care permite traficului ilegitim să fie trimis sau de la server sau de la client. De exemplu, un atacator care a lansat un atac $MITM$ contra unui client care comunică prin Telnet cu un sistem UNIX poate face serverul să gândească că primeşte de la client comanda **rm-rf*** (comandă UNIX de a şterge toate fişierele), când de fapt acesta nu a lansat aşa ceva.
  De asemenea, sesiunea poate fi încheiată de atacator.

- Înlocuieşte pachete având anumite secvenţe de biţi cu noi secvenţe alese de atacator.

10.10.2 Viruşi, Viermi şi Cai Troieni

Caracteristici:

1. **Membru al clasei/subclasei**: Composite.

2. **Exemple de implementări**: *SQL Slammer* (vierme), *Code Red* (vierme), *Melissa* (virus), *NetBus/Whack-a-Mole* (Troian).
3. **Preliminarii**: Variabile.
4. **Vulnerabilităţi**: Software şi utilizare.
5. **Utilizare tipică**: Variabilă
6. **Rezultate atac**: Variabilă (orice este posibil).
7. **Tipuri de atac folosite**: Citire, Manipulare, Spoof, Flood.
8. *OSI* **Layers**: 7
9. **Detectare**: IDS.
10. **Protecţie**: Aplicaţii de securitate şi software antivirus.
11. **Dificultatea detecţiei**: 3
12. **Facilitate de atac**: 4
13. **Frecvenţă**: 5
14. **Impact**: 4
15. **Clasificare generală**: 42

Trebuie făcută clară diferenţa dintre cele trei tipuri de atac: Viruşi, Viermi şi Troieni.

- Un **virus** este o bucată de cod maliţios care modifică altă piesă de software din sistem. În general este nevoie de o anumită intervenţie a utilizatorului (deschiderea unui attachment e-mail, citirea unui disc infectat etc). Un exemplu tipic este virusul *Melissa*. Acesta se transmite prin documente word trimise victimei prin attachment. Documentul word conţine un macro-cod maliţios care face ca virusul să se propage la primele 50 adrese din *Adress book*.

- Un **vierme (worm)** este un tool standard care infectează sistemele vulnerabile, iar acestea infectează la rândul lor alte sisteme. Un vierme infectează în general în mod automat, deşi uneori pentru lansare este nevoie de o acţiune a victimei (cum ar fi un click pe un atachment). De exemplu, *Code Red* care profită de o falie de securitate a serverului de indexare Microsoft (componentă a IIS – Internet Information Server) pentru a infecta sute de mii de sisteme. Din cauza propagării sale necontrolate, acest vierme poate avea şi unele efecte de *DoS* în anumite zone de Internet.

- Un **Cal Troian (Troian Horse)** este o aplicaţie care apare utilizatorului ca având o anumită funcţie, dar în realitate acţionând complet diferit. Ca exemplu, să luăm *NetBus/Whack-a-Mole*. Acesta – distribuit frecvent prin e-mail – apare utilizatorului ca un joc din baza Microsoft Windows (ca şi Solitaire de exemplu). În timp ce utilizatorul joacă acest joc (simpatic), aplicaţia instalează o listă de comutare pe un port TCP de nivel înalt, permiţând atacatorului să se conecteze la sistem şi să lanseze diverse atacuri, cum ar fi resetarea sistemului şi schimbarea proprietăţilor locale.

10.10.3 Rootkit

Caracteristici:

1. **Membru al clasei/subclasei**: Composite.
2. **Exemple de implementări**: *t0rn*.
3. **Preliminarii**: Acces la root.
4. **Vulnerabilităţi**: Software, configurare şi utilizare.
5. **Utilizare tipică**: Ascunde victimei prezenţa unui atacator.
6. **Rezultate atac**: Variabilă (orice este posibil).
7. **Tipuri de atac folosite**: Citire, Manipulare.
8. *OSI* **Layers**: $3 - 7$
9. **Detectare**: *Chkrootkit* şi $HIDS$.
10. **Protecţie**: Aplicaţii de securitate.
11. **Dificultatea detecţiei**: 4
12. **Facilitate de atac**: 2
13. **Frecvenţă**: 4
14. **Impact**: 4
15. **Clasificare generală**: 36

Rootkit permite atacatorilor să-şi ascundă prezenţa pe o maşină pe care deja au compromis-o. De exemplu, să presupunem că un atacator a compromis un host Linux. Atunci *t0rn* îi va permite să facă următoarele acţiuni:

1. Să dezinstaleze *syslogd*.
2. Să-şi stocheze parola pentru programe de tip Troian în */etc/ttyhash*.
3. Să instaleze o versiune troianizată a *sshd*, configurată pentru a urmări un anumit port al intrusului.
4. Să ascundă nume de fişiere, de procese etc.
5. Să înlocuiască cu copii troianizate fişierele binare */bin/login, /sbin/ifconfig, /bin/ps, /usr/bin/du, /bin/ls, /bin/netstat, /usr/sbin/in.fingerd, /usr/bin/find, /usr/bin/top*.
6. Să instaleze o parolă *sniffer, parser sniffer log file*, şi *system log file*.
7. Să încerce să conecteze *telnet, shell* şi *finger* în $etc/inetd.conf$ ştergând toate caracterele # de început de linie pentru comentarii.
8. Să restarteze */usr/bin/inetd*.
9. Să pornească *syslogd*.

După ce realizează aceste deziderate, atacatorul poate rula programe din sistem fără ca victima să aibă idee că sistemul ei este compromis.

Rootkit este o modalitate foarte eficientă pentru intruşi de a-şi ascunde prezenţa într-o reţea infectată. Detectarea lui este extrem de dificilă. Există un utilitar, *Chkrootkit* ([18]) care permite unui administrator să detecteze unele tooluri din rootkit în faza de execuţie.

10.10.4 Remote Control Software

Caracteristici:

1. **Membru al clasei/subclasei**: Composite.
2. **Exemple de implementări**: *Back Orifice 2000* ($BO2K$).
3. **Preliminarii**: Variabile.
4. **Vulnerabilităţi**: Software, configurare şi utilizare.
5. **Utilizare tipică**: Controlează sistemul victimei dintr-o locaţie externă.
6. **Rezultate atac**: Variabilă (orice este posibil).
7. **Tipuri de atac folosite**: Citire, Manipulare, Spoof.
8. OSI **Layers**: $3 - 7$
9. **Detectare**: IDS.
10. **Protecţie**: Aplicaţii de securitate, software antivirus.
11. **Dificultatea detecţiei**: 4
12. **Facilitate de atac**: 4
13. **Frecvenţă**: 3
14. **Impact**: 4
15. **Clasificare generală**: 37

Controlul exterior de software nu este întotdeauna o slăbiciune a sistemului; el este utilizat adesea de multe ori în scopuri legitime. Multe sisteme de operare includ posibilitatea de a muta controlul înafara sistemului, ca o facilitate pentru inginerii IT.

Metoda poate fi folosită însă şi de atacatori care doresc să preia de la distanţă controlul unui sistem. O metodă uzuală constă în trimiterea unui mesaj prin e-mail victimei, care conţine în attach un soft de preluare exterioară a controlului (troianul *NetBus* este un exemplu elocvent). Odată rulat, procesul se auto-ascunde în sistem prin diverse mijloace – de exemplu redenumind procesul pe care îl execută. Atacul poate fi pasul iniţial al unei palete largi de alte atacuri; lansarea unui atac $DDoS$ poate începe cu preluarea controlului unui număr mare de sisteme. Portul pe care lucrează este adesea specificat de utilizator, iar headerul pachetelor poate fi criptat.

Un tool de preluare a controlului foarte cunoscut este *Back Orifice 2000* sau $BO2K$, care poate fi descărcat de la adresa [17]. $BO2K$ permite comunicarea între client şi server criptată cu sistemul AES şi permite atacatorului să realizeze următoarele:

- Să blocheze maşina.
- Să captureze toate cheile din sistem.
- Să ruleze un fişier *.wav* sau să arate un mesaj pe monitor.
- Să aducă alte atacuri prin plug-in-uri.
- Să caute şi să transfere sistemul de fişiere locale.

- Să editeze regiştrii.

Toate acestea pot fi executate printr-un port specificat de utilizator folosind UDP, TCP sau chiar $ICMP$. Odată sistemul infectat, el devine "*zombie*" (**bot**): atacatorul poate face orice cu el, ca şi cum ar fi în faţa ecranului.

10.11 Concluzii

Tip de atac	(11)	(12)	(13)	(14)	**Rating**
Buffer Overflow	4	3	5	5	45
Identity Spoofing	4	3	4	5	42
War dialing/driving	5	4	3	5	42
Virus/Vierme/Cal troian	3	4	5	4	42
Acces direct	2	5	5	3	39
Remote control software	4	4	3	4	37
Verif/scan	4	5	5	2	37
Rootkit	4	2	4	4	36
Sniffer	5	5	3	3	36
Application flooding	3	5	5	2	36
UDP Spoofing	5	4	3	3	34
Aparate false	3	2	2	5	33
Aplicaţii web	3	3	4	3	33
Baleiere date	5	4	5	1	32
Man-in-the-middle	4	2	1	5	31
$DDoS$	2	2	3	4	31
TCP Soofing	5	1	1	5	30
Redirectare ARP	3	4	1	4	30
$TCP\ SYN$ flood	3	5	3	2	30
IP spoofing	3	4	5	1	30
Redirectare IP	2	2	2	4	28
Smurf	2	4	2	3	28
Redirectare transport	4	3	2	3	28
MAC flooding	3	5	1	3	28
MAC spoofing	3	5	1	3	28
Atacuri pe reţea	2	3	2	3	26
Redirectare STP	3	3	1	2	20

Capitolul 10 a efectuat o scurtă trecere în revistă a principalelor tipuri de atac existente asupra reţelelor wireless, şi cele mai uzuale riscuri la care sunt expuse sistemele conectate la Internet. Se încearcă o clasificare a lor după modul de acţiune şi după efecte. Aprecierea după dificultatea detecţiei (11), uşurinţa de utilizare (12), frecvenţa (13) şi impactul (14) atacurilor trebuie actualizată permanent, odată cu dezvoltarea atât a softurilor de securitate cât şi a noilor tipuri de atacuri care apar. Vom relua sub o formă extinsă cele mai cunoscute atacuri întâlnite.

Cred că este utilă o recapitulare a atacurilor prezentate, ordonate crescător după ratingul de periculozitate determinat în funcţie de cele patru criterii amintite. Tabelul prezentat mai sus realizează acest lucru.

Bibliografie

[1] Atanasiu, A – *Securitatea informaţiei, vol. 2 (Protocoale de securitate)*, Ed. Infodata, Cluj (2009)

[2] Camp, L.J., Wolfram, C. – *Pricing Security*, *www.cert.org/research/isw/isw*2000/54.*pdf*

[3] Conway, S. - *Network Security Architectures*, Editura Cisco Press (2005)

[4] Fyodor – *Remote OS detection via TCP/IP Stack FingerPrinting*, http://www.insecure.org/nmap/nmap-fingerprinting-article.html

[5] Howard, J. – *An Analysis of Security Incidents on the Internet, 1989-1995*, PhD Thesis, www.cert.org/research/JHThesis/Start.html.

[6] *Smashing the Stack for Fun and Profit*, Seminar Aleph One, $http://insecure.org/stf/smashstack.html$.

[7] *Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection*, $http://insecure.org/stf/secnet\_ids/secnet\_ids.htlm$.

[8] Paketto Kereiretsu – $http://www.doxpara.com/read.php/code/paketto.html$

[9] $http://www.w3.org/Security/Faq/wwwsf4.html$.

[10] $http://www.cgisecurity.com/articles/xss-faq.shtml$.

[11] $http://www.dcphonehome.com/$.

[12] `http://monkey.org/~dugsong/dsniff/.`

[13] `http://ettercap.sourceforge.net/.`

[14] $http://winfingerprint.sourceforge.net/aptoole.php$

[15] $http://monkey.org/\ dugsong/fragroute/$

[16] $http://www.atstake.com/research/tools/network\_utilities/$.

[17] $http://bo2k.sourceforge.net$.

[18] $http://www.chrootkit.org$.

[19] $http://www.wardriving.com$