# CONSIDERAȚII ASUPRA ATACURILOR CIBERNETICE, EXECUTATE ÎN CONTEXTUL COMUNICAȚIILOR PRIN REȚEA

**Ion Alexandru Marinescu** **Dragoş Nicolau** **Lidia Băjenaru**

ionut@ici.ro dragos@ici.ro lidia.bajenaru@ici.ro

Institutul Național de Cercetare-Dezvoltare în Informatică - ICI Bucureşti

**Rezumat:** Inițial, în perioada de început a Internetului, comunicațiile confereau siguranță. De mulți ani încoace însă, acest lucru nu se mai întâmplă, pentru că şi-au făcut apariția atacurile informatice. În principiu, acestea constau fie în emiterea unui flux foarte intens de cereri către o maşina server victimă pentru blocarea directă sau indirectă a traficului, fie în injecții de cod rău-intenționat în speranța că programul cărora le este destinat (aplicație client infectată de server sau aplicație server) va executa de bună credință, în acest fel furând sau viciind informație. În lucrarea de față vom prezenta principiul de funcționare al câtorva dintre cele mai răspândite şi periculoase atacuri la distanță, precum şi modalități de prevenire.

**Cuvinte cheie:** comunicații prin rețea, atac cibernetic, server, injecții rău-intenționate.

**Abstract:** Within the early years of the Internet Era, network communications used to enjoy a considerable level of operating safety. However, over the past several years, the advent of increasingly potent and frequent cyber attacks have succeeded to weaken and even erode this very favorable state of affairs. Cyber attacks are redoubtable enemies of network security and occur either- in the form of an extremely dense flow of requests released toward a server machine for the purpose of direct or indirect data-traffic obstruction or- in the form of stealth insertion of malicious program code meant to be interpreted/executed in full confidence by the targeted application (client side-type, client side-type infected via server-side, server-side) and thus to enable the theft or the partial or total destruction of valuable pieces of data. The present paper is designed to describe the functional principles of the most widely spread remote-cyber attacks and also to present the defensive measures that must be applied to achieve the goal of preventing/annihilating cyber aggressions.

**Keywords:** network communications, cyber-attack, server, malicious injection.

## 1. Introducere

Securitatea vehiculării datelor în format electronic este considerată în momentul de față un element critic al societății informaționale.

Este tot mai evident faptul că incidentele cu cauze "naturale" sau provocate din rețelele şi sistemele informatice produc prejudicii din ce în ce mai mari, iar măsurile de securitate devin din ce în ce mai complexe, fiind gândite să conducă la stăvilirea fenomenului. Prin urmare, una din direcțiile importante de acțiune constă în stabilirea unui cadru legal cît mai cuprinzător, corelat cu cele mai noi tehnologii destinate securității mediului virtual la nivelul societății (producători, comercianți, autorități, utilizatori etc.) pentru a obține eficiență maximă.

Proiectarea şi întreținerea unui sistem de securitate care să fie capabil să prevină sau să stopeze atacuri tot mai variate, înainte de a produce prejudicii, trebuie să țină pasul cu o tot mai complexă şi dezvoltată tehnologie a atacurilor, fapt pentru care se impune cunoaşterea, sincronizarea şi armonizarea mecanismelor legislative din state partenere, cu scopul imediat de a creşte nivelul de încredere în sistemele on-line de afaceri sau administrative, acestea din urmă constituind, în ultimă instanță, o dimensiune din ce în ce mai importantă a unei dezvoltări economice tot mai rapide, în contextul globalizării schimbului de mărfuri şi servicii.

Construit din interconectarea mai multor rețele, Internetul s-a construit şi dezvoltat rapid într-un mediu care conferea inițial siguranță. Ulterior s-a dovedit că Internetul este nesigur şi este folosit ca suport pentru inițierea de atacuri la distanță. Dezvoltarea rapidă a acestuia a făcut ca anumite standarde necesare bunei funcționări a Internetului să fie rapid proiectate, testate şi implementate. Ingeniozitatea atacatorilor avea să fie pusă la încercare o dată cu implementarea primelor măsuri de securitate la nivel local şi la nivel rețea. Succesele contorizate de cele două tabere aflate de o parte şi de alta a imaginarei linii de demarcație între bine şi rău au fost folosite

pentru atingerea scopului propus. De partea binelui, succesele au dus la crearea unor politici de securitate mai eficiente, la implementarea de noi tehnologii şi metode defensive sau de identificare a atacatorilor.

Trebuie menţionat faptul că industria software este domeniul de activitate care are de departe nu doar cea mai mare viteză de evoluţie şi diversificare, ci şi cea mai mare uşurinţă de participare la fenomenul internaţionalizării (atât în ceea ce priveşte dezvoltarea de produse soft, cât mai ales, în ceea ce priveşte utilizarea lor).

Prin urmare, pentru dezvoltarea şi adaptarea soluţiilor bazate pe cele mai recente tehnologii de securitate cibernetică, trebuie cunoscute tipurile de atac.

În continuare ne vom ocupa de prezentarea principalelor tipuri de atacuri cibernetice, executate în contextul comunicaţiilor prin reţea.

## 2. Atacuri de tip Denial of Services (DoS)

Denial of Services ( = 'împiedică/opreşte serviciul' – lb.engleză) presupune o acţiune care împiedică utilizarea normală a reţelelor, sistemelor sau aplicaţiilor prin consumarea resurselor cum ar fi unităţi centrale de procesare, memorie, lăţime de bandă, spaţiu de stocare etc. Cele mai dese atacuri de tip *DoS* sunt cele asupra serverelor de aplicaţii Internet, prin care atacatorii fac aceste situri indisponibile utilizatorilor. Câteva exemple de atacuri de tip DoS ar fi următoarele:

- *utilizarea în întregime a lărgimii de bandă prin generarea artificială de volum mare de trafic pentru a bloca traficul util.* Trimiterea către un server de pachete TCP/IP malformate pentru a bloca sistemul de operare. (TCP/IP = Transport Control Protocol/Internet Protocol = 'Protocol pentru controlul transportului, bazat pe Protocolul Internet' – lb.engleză. În principiu, acest protocol înseamnă o funcţionalitate – foarte concret, instrucţiuni aflate în fişierele DLL de sistem - care asigură trimiterea unor pachete de date prin reţea "cu confirmare de primire", rezultând o comunicare cu aşteptări şi eventuale retrimiteri, însă cu asigurarea transmiterii ireproşabile a datelor. Fiecare pachet trimis – cuantă de informaţie vehiculată prin reţea - are adăugată meta-informaţie de identificare, peste meta-informaţia adăugată informaţiei utile de către instrucţiunile care constituie protocolul de internet IP. Evident, acest mod de dialog este adoptat de către ambele entităţi care comunică, odată ce au acceptat stabilirea unui astfel de canal de comunicaţie. Spre diferenţă, protocolul UDP – User Datagram Protocol, nu verifică succesul transmiterii, fiind astfel folosit la aplicaţiile de tip telefonie – cum ar fi Skype, sau la rularea secvenţială a unor filme în direct, pe internet – cazul Youtube, adică acolo unde viteza primează faţa de acurateţe. Desigur, în cazul unui download sau al poştei electronice, se impune stabilirea unui canal de tip TCP/IP. Menţionăm că protocolul IP (suprapus imediat peste cel fizic elementar, Ethernet) cuprinde instrucţiuni pentru conectare la distanţă şi trimiterea informaţiei prin porţionare în pachete "etichetate".);
- *trimiterea de cereri ilegale către o aplicaţie pentru a o bloca;*
- *trimiterea de cereri care necesită procesare puternică pentru a consuma resursele procesoarelor;*
- *stabilirea simultană pe un server a unui număr mare de sesiuni de logare astfel încât utilizatorii legitimi să nu mai poată iniţia sesiuni de logare noi – acest atac consumă memorie RAM, resurse procesor şi timp, pe serverul atacat;*
- *consumarea în foarte mare măsură a spaţiului de pe unităţile de stocare prin crearea de fişiere mari.*

Dintre toate atacurile de tip *DoS*, patru sunt cele care necesită o atenţie deosebită: *atacurile reflector, atacurile amplificator, atacurile DoS distribuite (DDoS)* şi *synfloods*. Acestea sunt prezentate pe scurt în continuare.

(1) Într-un atac de tip reflector, host-ul atacator (host = 'gazdă' – lb.engleză, maşină pe care

rulează un anume program sau script) trimite către un host intermediar un număr mare de cereri cu adresa sursă falsă (a host-ului atacat). Host-ul intermediar trimite host-ului atacat răspunsuri de bună credință la toate cererile sosite aparent de la acesta, blocându-l prin depăşirea capacității de procesare a răspunsurilor. Deoarece host-ul intermediar produce în mod neintenționat atacul, acesta se numeşte reflector. Este de menționat că, în timpul unui astfel de atac, host-ul reflector poate la rândul său să sufere un DoS, dacă propria lui capacitate este depăşită.

(2) La fel ca în cazul atacului reflector, atacul amplificator presupune trimiterea de cereri cu adresa sursă falsă către un host intermediar. Atacul reflector însă nu utilizează numai un host intermediar, ci o întreagă rețea de host-uri intermediare. Acest lucru se realizează prin trimiterea cererilor către toate host-urile dintr-o rețea (printr-o adresă de broadcast), astfel încât o multitudine de host-uri să trimită răspunsuri către adresa falsă indicată ca sursă de atacator. În acest caz, se produce un efect de amplificare care sporeşte semnificativ şansele de blocare a sistemului atacat.

(3) DoS distribuit – DdoS este o variantă şi mai performantă, însemnând compunerea mai multor atacuri de tip DoS.

În general, sistemele/rețelele din majoritatea organizațiilor au la dispoziție lățimi de bandă mari, atât de mari încât atacuri DoS efectuate de pe un singur host, fie ele prin reflector sau amplificate, nu pot provoca efecte semnificative asupra funcționării acestora. Din acest motiv, atacatorii întreprind atacuri în mod distribuit, de pe mai multe host-uri. Dacă sunt folosite suficient de multe host-uri atacatoare, se poate ajunge ca volumul total de trafic generat să sufoce sistemele. Atacurile DDoS folosesc în principal două elemente agenți, care se instalează pe host-urile compromise şi execută propriu-zis atacul şi manipulatori, care sunt software-uri de control al agenților. Prin aceştia din urmă, agenții sunt controlați şi li se indică ce să atace, când şi cum. Într-un atac DDoS, atacatorul poate folosi de la sute până la mii de agenți.

(4) Un atac synflood ( flood = 'inundație' – lb.engleză) apare când atacatorul inițiază un număr mare de conexiuni TCP într-un timp foarte scurt prin trimiterea de pachete pentru a excita primirea de răspunsuri preliminare SYN, fără însă a re-trimite la server confirmarea ACK, lăsând astfel conexiunea deschisă, practic în aşteptare. Sistemele de operare în general permit un număr mic de conexiuni în aşteptare (conexiune incompletă). Dacă un atacator inițiază 100 de astfel de (pseudo) conexiuni TCP pe un anumit serviciu/port şi nu stabileşte niciuna dintre ele, sistemul de operare atacat nu va mai avea resurse disponibile pentru inițierea de noi conexiuni pentru acel serviciu/port până nu sunt închise conexiunile în lucru, din cauza nefinalizării procesului de stabilire a conexiunii – lucru dictat de însuşi protocolul TCP. Această operație poate dura aproximativ un minut, timp în care practic are loc o indisponibilitate a serviciilor (DoS). Dacă atacatorul continuă inițierea de conexiuni, atunci efectul blocării creşte, generând o blocare completă a resurselor sistemului de operare şi, în speță, a rețelei / sistemului / serviciilor organizației.

**Măsuri de prevenire**

Dintre măsurile de prevenire consemnăm:

- *utilizarea comenzii de verificare a drumul IP-ul vizitator, executate pe router în amontele conexiunii. Dacă IP vizitator nu-şi regăseşte traseul inițial în tabela consacrată Cisco Express Forwarding din memoria router-ului, înseamnă că este IP de fațadă, deci va fi respin;.*
- *filtrarea gamelor de IP-ri acceptate;*
- *asocierea grupurilor de IP cu grupuri de acces;*
- *configurarea limitării cadenței cu care sosesc pachetele SYN;*
- *analizarea frecvenței cu care sosesc cereri de la un anume IP;*
- *jurnalizarea IP-urilor suspecte şi raportarea lor către o autoritate CERT.*

Măsurile de prevenire se iau nu doar la nivel de router, ci şi la nivel de serviciu sau aplicație care ascultă pe un anumit ***:port***, pe o maşină server.

## 3. Atacuri de tip cod răufăcător (malicious code)

Codul maliţios reprezintă un program care este inserat pe ascuns într-un alt program în scopul de a provoca daune (distruge date, rulează programe distructive / intruzive, compromite securitatea sistemului) victimei fără ştirea utilizatorului. Atacurile prin cod maliţios pot fi împărţite în cinci mari categorii: *viruşi, cai troieni, viermi, cod mobil şi combinaţii*.

Virusul este este un cod maliţios ascuns, ataşat de fişiere de date sau executabile şi destinat, odată ce a avut şansa lansării primordiale în execuţie, să pornească automat la startarea maşinii, să producă daune şi să se autoreplice. Un virus este un mini executabil în cod maşină, **injectat** fie într-un executabil gazdă, fie într-un fişier de date gazdă (documente, filme, imagini). În acest ultim caz, este total inofensiv. Cu toate acestea, un fişier MS Office *.doc sau *.xls poate deveni periculos. Iată cum:

- *atacatorul injectează cod maşină undeva într-o secţiune a documentului;*
- *creează cu VBA un Macro (suită de subrutine Visual Basic, INTERPRETABILE, NU executabile), declanşabil la evenimentul Document_Open (deschidere document);*

```
Private Sub Document_Open()
    ' aici se inserează codul Visual Basic INTERPRETABIL
    ' care va lansa in execute codul maşina EXECUTABIL(virus)
    ' . . .
End Sub
```

- *funcţiile de tip Visual Basic din Macro-ul acum lansat în interpretare fac o copie a documentului, citesc zona de adrese unde locuieşte codul maşină, îl salvează temporar pe HDD ca .exe, după care îl lansează în execuţie cu o simplă comandă de shell. Virusul se poate scrie în sectorul de Boot sau îşi creează propria autodeclanşare prin zona de start-up din regiştri.*

**Măsuri de prevenire**

Instalare de programe antivirus cît mai performante şi periodic actualizabile. Microsoft a instalat în noile generaţii de Office opţiune de dezactivare automată (reversibilă, pe riscul clientului) a Macro-urilor.

## 4. Atacuri de tip Acces Neautorizat (Unauthorized Access)

Un atac de tip *acces neautorizat* apare atunci când un utilizator/atacator obţine acces la resurse la care în mod normal nu are acces. Accesul neautorizat este în general obţinut prin exploatarea unor vulnerabilităţi ale sistemului de operare sau ale aplicaţiilor, prin intrarea în posesia unor parole şi nume de utilizator sau prin inginerie socială. Atacatorii pot obţine în primă fază acces limitat prin exploatarea unor vulnerabilităţi şi pot utiliza acest tip de acces pentru a folosi alte vulnerabilităţi care aduc treptat niveluri de acces sporit. Exemple de acces neautorizat includ:

- *compromiterea contului de administrare pe un server de e-mail;*
- *compromiterea unui server de Web;*
- *spargerea de parole;*
- *copierea de baze de date cu numere de carduri de credit;*
- *accesul neautorizat la date confidenţiale, incluzând state de plată sau date medicale etc.;*
- *folosirea de programe sniffer (sniffer = 'adulmecător' – lb.engleză; este un program executabil care captează/ascultă/citeşte şi eventual memorează tot traficul de reţea – date, adrese IP ale surselor, adrese IP ale destinaţiilor – al maşinii unde este instalat) pe host-uri pentru a captura nume de utilizatori şi parole;*

- *distribuirea de software sau fişiere media piratate prin acces neautorizat pe servere FTP;*
- *folosirea fără permisiune a unor staţii de lucru rămase logate şi nesupravegheate.*

Vom menţiona atacurile de compromitere a unui server Web, referindu-ne la injecţii Javascript, injecţii SQL şi interpunerea (Man-In-The-Middle).

(1) Injecţiile Javascript reprezintă cod interpretabil de tip Javascript, trimis către server de către un utilizator rău intenţionat, pentru a fi executat ulterior pe maşinile client ale altor utilizatori ai aceleiaşi aplicaţii Web. Un caz concret de infectare cu astfel de injecţie este un sit de tip blog ( = 'jurnal electronic' - lb.engleză), unde fiecare vizitator are posibilitatea să insereze ce text doreşte (de exemplu, un comentariu la un articol, sau comentariu la comentariu etc.) într-o casetă de editare (control HTML de tip <input>) prezentă în pagină, destinată acestui scop de către proiectantul aplicaţiei. După definitivarea textului, utilizatorul va apăsa butonul de trimitere (submit), în acest fel textul introdus fiind trimis către serverul care găzduieşte aplicaţia, textul odată ajuns pe server, fiind, evident, memorat în baza de date ataşată aplicaţiei (să presupunem că este inserat în tabela "Comentarii", la articolul 15, al autorului "Aut3", pe calea "5-2-7"; în acest exemplu de memorare, textul curent este al 7-lea comentariu la al 2-lea comentariu la al 5-lea comentariu la articolul 15 al autorului "Aut3") .

Un utilizator rău intenţionat, însă, va insera următorul text:

**`<script> alert('Un exemplu de injectie JS !'); </script>`**,

care va fi memorat ca atare în baza de date.

În momentul în care oricare alt vizitator va vizita pagina cu articolul 15 al autorului "Aut3", în zona de afişare (de obicei, controlul HTML de tip **`<div>`**) va fi încărcat din baza de date inclusiv textul răufăcător. Odată ajuns pe partea de client, acest script va fi automat interpretat-executat de către un browser cu anumite vulnerabilităţi. Codul maliţios (cod Javascript) poate încerca să trimită fişierele de tip cookie ale sesiunii, către un sit maliţios, prin comenzi care: creează dinamic un element **`<div>`** sau **`<input>`** în cadrul unui **`<form>`** invizibil temporar, care va fi trimis (submited) la destinaţia maliţioasă. Elementul **`<div>`** va conţine textul document.cookie (valoare care include identificatorul sesiunii curente, iar câmpul action la elementul form va conţine destinaţia, de exemplu "http://sitfraudulos.org". Trimiterea se va face prin comanda **`formul.submit();`**. Reamintim că un cookie ( = 'fursec' – lb.engleză) este un mic fişier text auxiliar, în care browserul ţine informaţii la îndemână, cum ar fi identificatorul de sesiune (string unic generat pe partea de server), pe care browserul îl trimite la fiecare cerere către site-ul asociat, pentru a scuti utilizatorul de noi autentificări prin nume şi parolă. Identificatorul de sesiune funcţionează ca un "paşaport de încredere" în dialogul client-server, pe durata unei sesiuni odată stabilite prin procesul de autentificare.

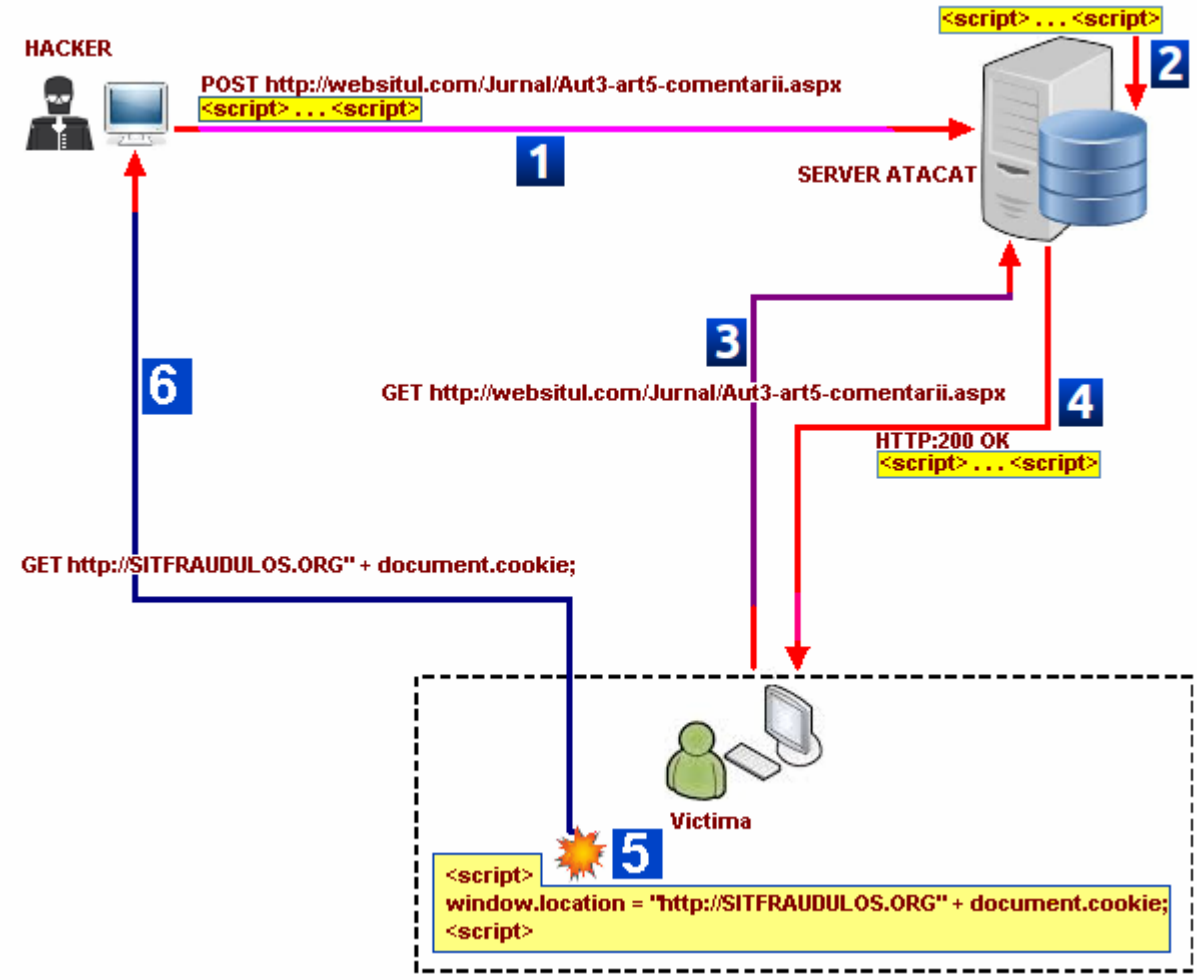
Browserele actuale sunt "vaccinate" împotriva executării de astfel de acţiuni, cum ar fi cele din exemplele de mai jos:

- *nu permit citirea de cookie sau date provenind din alt domeniu;*
- *nu permit executarea unui cod Javascript plasat in bara de URL;.*
- *nu permit executarea unui cod Javascript inserat dinamic în controale ascunse;*
- *nu permit citirea/trimiterea de informaţie de pe/către alt domeniu (cross site scripting) etc.*

Din nefericire, există totuşi modalităţi de a insera cod Javascript maliţios, executabil de către browser, de aceea este puternic recomandat ca securizarea să nu fie lăsată doar pe umerii aplicaţiei client (browserul), ci să revină în primul rând în sarcina codului aplicaţiei Web, pe partea de server. **Măsura de prevenire** constă în analizarea datelor care ajung pe server la aplicaţia Web, şi neutralizarea lor, în caz de suspiciune. Mai jos ilustrăm această manevră, executată în asp.net C#: în baza de date nu se va scrie textul s ca atare, ci se va scrie **`HttpUtility.HtmlEncode(s);`**. Prin urmare, dacă de exemplu, textul ar conţine o tentativă de cod maliţios, cum ar fi

**`<script> {un cod} </script>`**, sau **`javascript:{un cod}`**,

atunci în baza de date s-ar scrie un text care a suferit modificări în zona de identitate de tip HTML, adică `<script> {un cod};><&#47;script>`, sau, de exemplu, `_javascript_::`, ceea ce va însemna eroare de sintaxă Javascript ! În acest caz, codul răufăcător este neutralizat, devenind imposibil de *interpretat de către navigatorul client*.




**Schema unui atac de tip injecție Javascript**

Figura de mai sus redă schema unui atac de tip injecție Javascript, compusă din 6 paşi [1]. În această imagine, GET reprezintă o cerere HTTP de aducere a unei resurse (fişier) de pe server pe partea de client, iar POST reprezintă o cerere HTTP de scriere a unei anumite informații (aici, atacul sub forma de <script> {un cod} </script>), memorarea informației fiind executată de către modulul asociat paginii înscrise în cerere. Pasul (2) este executat imediat, pe server, după primirea cererii POST, iar pasul (4), adică "infectarea" cu cod malițios, este executat în momentul în care victima navighează pe site-ul atacat; (5) este executat pe client, automat după(4).

(2) Injecțiile SQL reprezintă cod SQL malițios care, inserat pe partea de client în casete de text şi trimis pe serverul aplicației, odată interpretat de motorul SQL al Sistemului de Baze de Date, execută operații răufăcătoare.

Redăm mai jos scenariul unei astfel de operații, pentru cazul unei aplicații Web scrise în asp.net C# şi servită de către ***Internet Information Service***.[4]

- *clientul rău intenționat accesează pagina de start a aplicației Web atacate;*
- *presupunând că este deja înregistrat, utilizatorul trebuie doar să se autentifice, prin nume şi parolă, aşa că va apăsa butonul de trimitere, după completarea celor 2 câmpuri;*
- *informația ajunge la aplicația țintă, unde este prelucrată de metoda OnInit(), asociată paginii de start;*

- *în cadrul acestei metode, valorile asociate numelui şi parolei (adică setările concrete) sunt extrase şi apoi inserate în textul de comandă SQL;*
- *acesta din urmă este trimis către interpretare serverului de baze de date;*
- *textul SQL este gândit să verifice existența în tabelul de utilizatori a perechii {nume: parola}, însă, fiind viciat de setarea malițioasă, poartă în el pericolul atacului asupra bazei de date;*
- *textul modificat este interpretat de către serverul de baze de date şi atacul este produs.*

Presupunem că la autentificare pe pagina Web, răufăcătorul nu setează un nume şi o parolă ”onestă”, ci setează nume = “Oarecare” şi parola = “**’;drop table UTILIZ;--**”.

Efectul, explicitat mai jos, este transformarea instrucțiunii SQL, gândite să facă o simplă citire, în două instrucțiuni valide (prima, inofensivă, cea de-a doua, periculoasă) plus un neutru segment de comentariu.

```
SELECT id, nume, rol FROM UTILIZ WHERE nume = 'Oarecare'
AND parola = ‘’; drop table UTILIZ; --’ ORDER BY nume DESC;
```

Prima instrucțiune, perfect corectă dpdv al sintaxei SQL, va fi:

```
SELECT id,nume,rol FROM UTILIZ WHERE nume = 'Oarecare' AND parola =
‘’;
```

A doua instrucțiune (atacul efectiv), impecabilă dpdv al sintaxei SQL, va fi:

```
drop table UTILIZ;
```

Zona comentată va fi:

```
--’ ORDER BY nume DESC;
```

În textul atacator, roluri sunt stabilite după cum urmează: (1) “**’**”apostrof = va delimita prima instrucțiune; (2) “**;**” = va semnala începerea instrucțiunii de atac propriu-zise; (3) atacul efectiv, în acest exemplu, tabelul numit “UTILIZ” fiind şters, dacă există; (4) “**--**“ va neutraliza un eventual rest al comenzii inițiale, comentându-l. Pentru acest caz ilustrativ, sintetizăm în figura de mai jos decelarea pe componente a anatomiei de principiu a unui atac de tip injecție SQL.

| 1 | 2 | 3 | 4 |
|---|---|---|---|
| ’ | ; | drop table UTILIZ ; | -- |

**Componentele de principiu ale unui atac de tip injecție SQL**

**Măsuri de prevenire.** Desigur, acest exemplu arată doar principiul de lucru, în locul comenzii care şterge un tabel putându-se insera orice instrucțiune SQL, care ar putea încerca ştergerea/înlocuirea unor date, citirea/schimbarea unor date, citirea/schimbarea unor nume de câmpuri sau nume de tabele, citirea/schimbarea unor privilegii de administrare intrinseci bazei de date atacate etc. Cu toate că succesul unei injecții se bazează şi pe şansă, configurarea şi dezvoltarea unei aplicații sigure pe partea de server rămân imperios necesare. Pentru a contracara injecțiile SQL, se poate recurge la două metode:

- *restricționarea acordării de drepturi de acces pe baza de date utilizatorului de tip asp.net anonim, fie prin cod SQL, fie prin aplicația client a Sistemului de Gestiune Baze de Date;*
- *apelarea unei funcții de analiză-filtrare în modulele din aplicație care execută dialog cu baza de date.*[3]

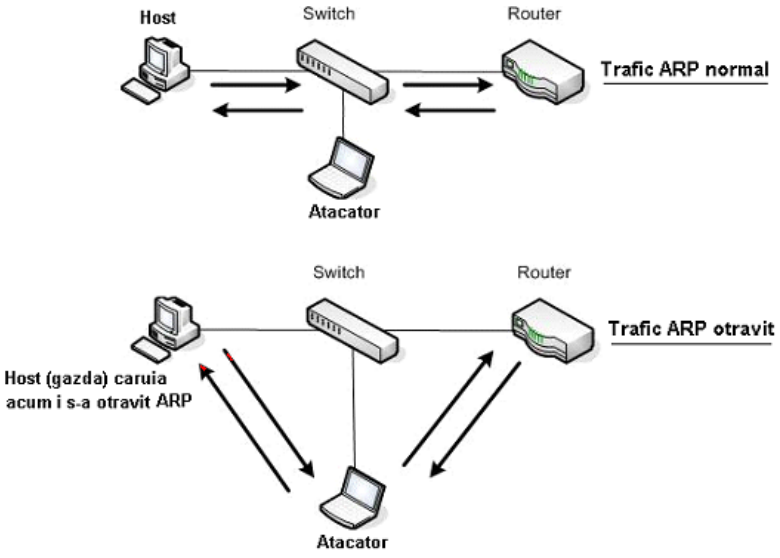
(3) Interpunerea pe traseu sau ‘Man-In-The-Middle’ – lb.engleză. (MITM) reprezintă un foarte periculos mod de atac asupra serverelor, cu implicații grave asupra utilizatorilor individuali. Într-un atac de tip MITM, atacatorul execută plasarea logică a propriului calculator într-un punct de intermediere între alte două maşini. Odată ajuns în această poziție, atacatorul poate lansa o serie de

atacuri foarte periculoase, din cauza posibilității de interceptare a mesajelor transmise între mașinile țintă. Există multe atacuri de tip MITM, astfel că ne vom concentra pe atacul asupra protocolului ARP, asupra protocolului DNS , asupra protocolului SSL și asupra sesiunii HTTP.

*(a) Protocolul ARP* este un protocol nivel 3 utilizat pentru a traduce adrese IP (ex: **`192.168.1.1`**) în adrese fizice ale plăcii de rețea sau adrese MAC (ex: **`0fe1.2ab6.2398`**). Când un dispozitiv încearcă să acceseze o resursă de rețea, mai întâi va trimite cereri către alte dispozitive solicitând adresa MAC asociată adresei IP pe care dorește să o contacteze. Apelantul va păstra asocierea IP-MAC în cache-ul ( = 'stoc de rezervă' – lb.engleză, prin urmare informație memorată pe hard disk pentru acțiuni repetitive ulterioare pe aceleași resurse) ARP, pentru a mări viteza de conectare la aceeași adresă IP. Atacul apare atunci când o mașină execută o cerere către celelalte mașini pentru a afla adresa MAC asociată cu o adresă IP. Atacatorul va răspunde trimițând pachete false ce identifică adresa IP solicitată cu propriul MAC și astfel, scurtcircuitează asocierea reală IP-MAC ce poate proveni de la altă mașină. Acest atac se numește otrăvire ARP(ARP poisoning) și este posibilă numai dacă atacatorul se află în același domeniu (acesta din urmă fiind definit de către o adresa IP și o mască subnet, de exemplu **`192.168.1.1 / 255.255.255.0`**).

Vom explicita rolul protocolului ARP printr-un exemplu concret. Protocolul ARP constă în vehicularea celor două pachete, adică o cerere ARP și un răspuns ARP. Scopul solicitării și răspunsul sunt necesare pentru a localiza hardware adresa MAC asociată cu o adresă IP dată, astfel ca traficul să poate ajunge la destinație prin rețea. Pachetul cerere este trimis la fiecare dispozitiv pe segmentul de rețea și spune: "Adresa mea de IP este **`XX.XX.XX.XX`**, și adresa mea MAC este **`XX: XX: XX: XX: XX: XX`**. Trebuie să trimită ceva la oricine are adresa IP **`YY.YY.YY.YY`**, dar nu știu ce adresa are hardware-ul (adică MAC). Cine are acest IP solicitat e rugat să trimită adresa!". Răspunsul va veni în pachete "Către aparatul de emisie, eu am adresa IP a **`YY.YY.YY.YY`** și adresa MAC **`zz: zz: zz: zz: zz: zz`**". Odată ce acest lucru este finalizat aparatul de emisie va actualiza tabelul de cache ARP și dispozitivele sunt capabili să comunice unul cu altul [2].

Otrăvirea protocolului ARP constă în exploatarea vulnerabilităților inerente ale ARP. Spre diferență de protocolul DNS, care acceptă doar setări dinamice sigure, ARP va accepta actualizări fără ezitare, oricând. Acest lucru înseamnă că orice mașină poate trimite către un host anume un pachet de răspuns, forțând hostul să facă actualizarea cache-ului ARP cu noua valoare. Trimiterea unui răspuns ARP fără solicitare prealabilă se numește "trimitere ARP gratuită". Când acestea se fac cu intenție rea, câteva pachete gratuite bine țintite sunt suficiente pentru a induce în eroare hostul atacat, făcându-l să creadă că realizează comunicare de bună credință cu un alt host, când în realitate el dialoghează cu un atacator care "stă și ascultă tot ce se trafichează".




**Schema de principiu a atacului de tip Otrăvire ARP**

**Măsuri de prevenire**

Protejarea contra acestui gen de atac (evident, provenit doar din propria subrețea) trebuie să fie profilactică. Desigur, administratorii sunt în mai mare măsură preocupați de atacurile externe şi în mai mică de cele interne, astfel că trebuie să aibă grijă să țină sub control instalarea oricărui nou dispozitiv suspect în rețea, sau modificările aduse unuia deja existent. Modalități tehnice concrete includ:

- *scanarea periodică a gazdelor din subrețea cu comanda consolă arp –a (pentru Windows). Acest executabil din sistem afişează cache-ul ARP sub formă tabelară, în perechile IP:MAC;*
- *utilizarea softurilor specializate în detectare de intruziuni (Snort, xARP);*
- *securitatea porturilor este o funcție disponibilă pe switch-urile de capăt. Se permite astfel numai dispozitivelor cu anumite adrese MAC conectarea la porturile switch-urilor, astfel că, în cazul unei maşini neautorizate, switch-ul poate acționa şi avertiza administratorul sau poate opri imediat portul respectiv.*

*(b) Protocolul DNS* ( = Domain Name Server = 'serverul cu nume de domenii' – Eng.) înseamnă funcționalitatea care transformă o adresă în format literal prietenos ("**`www.google.com`**") în adresa IP asociată (în acest caz, **`217.73.160.249`**). Acest lucru este necesar deoarece routerele şi alte dispozitive de rețea nu înțeleg formularea literală, ci doar pe cea cifrică (precum IP). În esență, atacul constă în înlocuirea frauduloasă, în asocierea {**`www.bancaoarecare.com :192.80.70.60`**} rezidentă în DNS atacat, a componentei IP real cu IP-ul site-ul răufăcător, astfel ca toate cererile către **`www.bancaoarecare.com`** să fie direcționate către **`www.sitatacator.org`**. Acest gen de atac poartă denumirea de "trişarea / păcălirea DNS" (DNS spoofing).

Asocierile {**`adresă literală : adresă IP`**} sunt memorate în baze de date.

Aceste informații sunt comunicate clienților sau altor servere de tip DNS, dat fiind faptul că protocolul DNS funcționează pe principiul cerere/răspuns. Arhitectura structurii DNS având un grad ridicat de complexitate, mai jos vom ilustra pe scurt principiul de funcționare al protocolului DNS. De exemplu, un client dorind să rezolve un nume de domeniu trimite o cerere către serverul DNS. Conform protocolului, serverul trimite imediat un răspuns, astfel încât, din perspectiva clientului, singurele fluxuri de pachete sunt cererea şi răspunsul. Problema care se ridică este recursivitatea cererilor, deoarece internetul este structurat sub forma unui arbore de ierarhii. Prin procesul de recursivitate, servere DNS trimit cererea la servere DNS de nivel ierarhic superior, lucru perfect normal, dacă avem în vedere că un server anume memorează doar adresele gazdelor din propria rețea şi nu a celor existente în întregul Internet. Din acest motiv, un DNS intermediar se comportă el însuşi ca un client.

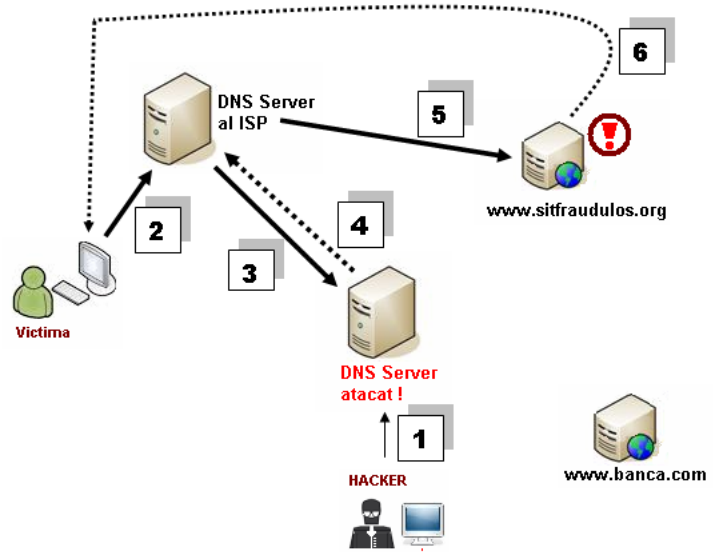
Trişarea DNS ( = 'spoofing') se face conform principiului arătat în continuare. Fiecare cerere obişnuită în cadrul unei navigări pe Internet conține un număr unic de identificare, pentru a o asocia cu răspunsul adecvat, acest lucru însemnând că, dacă atacatorul reuşeşte să se interpună pe traseu solicitării de tip DNS emise de serverul victimă, tot ce are de făcut este să creeze un pachet fals care conține numărul de identificare, pentru a păcăli victima să accepte respectivul pachet. Operațiunea răufăcătoare se face cu softuri specializate, cum ar fi Ettercap, şi cuprinde două etape principale:

- *otrăvirea cache-ului ARP a serverului victimă, pentru a deturna traficul său către situl malițios (host-ul răufăcător);*
- *când cererea de tip DNS ajunge la gazda malițioasă (host-ul răufăcător), acesta va retrimite pachetul cu număr de identificare falsificat.*

Ettercap, aplicație consolă, este în esență un ascultător de pachete care va folosi extensia dns_spoof plug-in pentru a executa atacul în propria subrețea.

Pentru aceasta, atacatorul va configura manual, pe propria maşină, fişierul text de evidenţă a DNS, unde va înscrie IP fraudulos către care vor fi direcţionate cererile de navigare către {**www.bancaoarecare.com**}. La acel IP va rula o aplicaţie web care va reproduce în mod fraudulos site-ul original. Ettercap va fi lansat în execuţie cu specificarea atacării întregii subreţele, făcând cele două operaţii, anume otrăvirea ARP şi trimiterea de pachete cu etichetă de identificare falsificată. În imaginea de mai jos este reprezentat principiul de funcţionare al atacului de tip DNS spoofing, unde:

- *(1) hackerul ( infractor informatic) execută infectarea;*
- *(2) clientul trimite cerere HTTP /www.banca.com;*
- *(3) ISP trimite cerere DNS către Serverul DNS imediat superior în ierarhie (cel atacat);*
- *(4) DNS infectat trimite IP sitului fraudulos, împreună cu eticheta de identificare falsificată;*
- *(5) ISP trimite cerere HTTP /www.sitfraudulos.org;*
- *(6) pagina frauduloasă ajunge pe maşina client.*




**Schema de principiu a atacului de tip DNS spoofing**

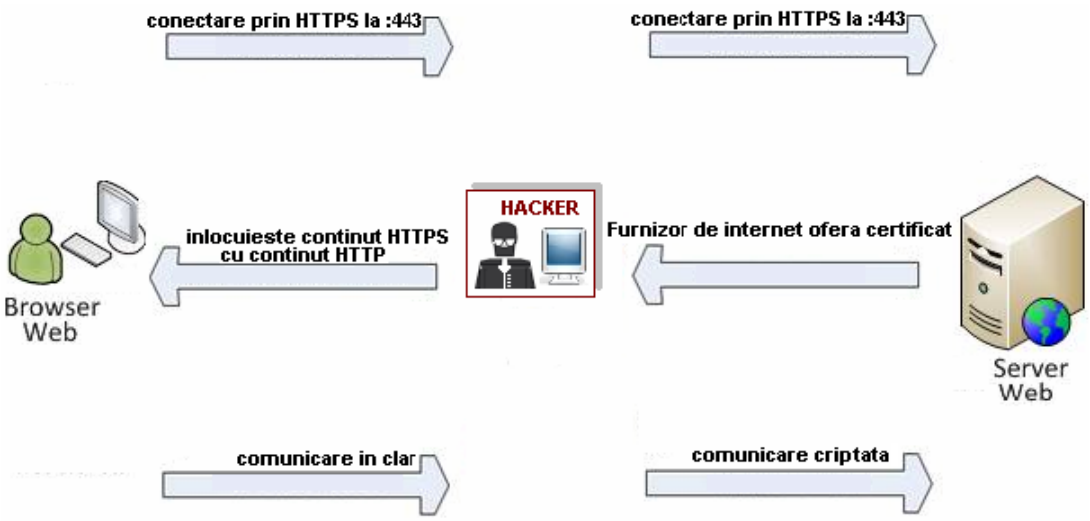
### Măsuri de prevenire

Mijloacele de contracarare sunt destul de dificile, deoarece acest atac este pasiv, esenţialmente. Atacul de tip DNS spoofing nu are simptomatologie, sesizabil fiind doar ceea ce se întâmplă clientului. Există totuşi câteva lucruri care se pot face:

- *securizarea maşinilor interne: în majoritatea zdrobitoare a cazurilor, acest atac este declanşat din interiorul subreţelei, prin urmare se impune atenţie la modificările apărute aici;*
- *a nu avea încredere în securizarea serverelor DNS superioare în ierarhia Internet;*
- *a se utiliza softul IDS, un instrument anti-intruziune;*
- *pe cât posibil, a se utiliza DNSSEC, o variantă securizată de DNS, care utilizează înregistrări cu semnătură: acesta este mecanism nou, în curs de desăvârşire, dar care a fost deja recomandat serverelor militare şi guvernamentale din SUA.*

(c) Protocolul SSL (Secure Socket Layer – 'nivel de comunicare securizat', sau TSL – Transport Layer Security – 'securizarea nivelului de transport' – lb.engleză) se referă la utilizarea criptării în transmisia de informație prin rețea. Acest protocol poate deveni suport pentru protocoale de aplicație securizate, cum ar fi SMTPS, IMAPS – pentru poşta electronică sau HTTPS – pentru utilizare Internet.

Atacul asupra protocolului SSL, considerat până acum câțiva ani drept imbatabil, se face propriu-zis prin atacare indirectă, deoarece victima a fost redirectată către HTTPS prin răspuns HTTP având codul 302. Ideea este atacarea drumului de la HTTP către HTTPS. Aceasta este posibil utilizând softul "SSLstrip", a cărui acțiune o vom ilustra în imaginea de mai jos.




**Schema de principiu a unui atac la protocol HTTP/SSL**

Funcționarea atacului se bazează pe următorii paşi:

- *interceptare trafic client server;*
- *când e întâlnită o cerere de tip HTTPS, sslstrip o înlocuieşte cu una HTTP şi memorează modificările;*
- *hackerul trimite le server certificatul, în numele clientului, acesta fiind acum victimă a unui act de "impostură" informatică;*
- *traficul este primit de la server şi retransmis clientului.*

În plus, atacatorul poate instala un sniffer de rețea şi memora toată informația schimbată (nume, parole, numere de conturi bancare etc.). Acesta este un atac de tip MITM produs înainte ca sesiunea încriptată să se stabilească. Procesul rulează cu succes din perspectiva serverului, care primeşte mesaje criptate. În ceea ce-l priveşte pe client, dacă este avizat, poate remarca imediat ca în bara de URL a browserului nu mai figurează protocolul HTTPS, ci HTTP! Cu toate acestea, se pot lua măsuri de prevenire a unui astfel de atac, pe partea de client, pentru că odată produs, atacul este cu greu detectabil pe server.

Pentru a fi executate din exterior, autorul atacurilor trebuie să dispună de instrumente soft performante pentru: (a) scanarea de la distanță a domeniului țintă (prin trimiterea de cereri variate – DNS, FPS, HTTP etc.) şi (b) penetrarea dispozitivelor software de protecție (firewall etc.)

Serverul nu are cum să sesiseze că dialoghează cu un interpus, şi nu cu clientul real.

## Măsuri de prevenire

- *asigurarea că se utilizează o conexiune SSL, lucru indicat de către bara de URL a*

*browserului;*

- *a se folosi conexiunea către bancă preferabil de la domiciliu, nu pentru că aici ar funcționa un calculator mai securizat, ci pentru că la domiciliu se conectează o singură mașină, în timp ce pe rețeaua companiei sunt conectate zeci sau sute de mașini (local sau la distanță !), ceea ce creşte riscul expunerii la un atac de tip MITM;*
- *pe partea de server, rămâne valabilă sugestia supravegherii modificărilor soft şi hard apărute în rețea, pentru că marea majoritate a atacurilor de tip MITM se execută din interior.*

## 5. Concluzii

Odată cu apariția punctelor de răscruce în utilizarea Internetului (e-Guvernare, e-Comerț, e-Banking, situri de schimb şi propagare a informației) a crescut corespunzător şi pericolul atacurilor cibernetice.

Atacurile vizează:

- *identitățile electronice;*
- *datele personale;*
- *informațiile bancare;*
- *informațiile sensibile (administrative, guvernamentale, militare), efectul colateral cel mai important fiind reticența individului sau a companiei în a utiliza serviciile e-Guvernării.*

Contracararea lor înseamnă o acțiune vitală pentru protejarea informației, ceea ce incumbă un permanent efort pentru studierea, înțelegerea şi păstrarea pasului cu mecanismele agresiunilor informatice, tot mai diversificate şi mai imaginative, în slujba cărora se pun instrumente software comode şi la îndemână.

Lucrarea de față şi-a propus să ofere o prezentare agregată a câtorva dintre cele mai redutabile (şi am spune interesante) tipuri de atac informatic (cele care vizează serverele), împreună cu explicitarea principiilor intime de lucru pentru fiecare atac studiat, pentru o mai rapidă înțelegere a funcționării inserându-se şi scheme explicative.

Indiferent de momentul apariției fiecărui atac adus în discuție, rămân permanent valabile şi obligatoriu de avut în vedere măsurile profilactice pe care trebuie să le aibă în vedere dezvoltatorii de aplicații şi servicii Web, dezvoltatorii de aplicații client, administratorii de sisteme de baze de date, administratorii de rețele şi de sisteme de unități de calcul.

## BIBLIOGRAFIE


1. **KALIN, JACOB Ş.A.:** A comprehensive tutorial on cross-site scripting, http://excess-xss.com/ ;
2. **SANDERS, CHRIS:** Understanding Man-in-the-Middle Attacks, 2010, http://www.windowsecurity.com/articles-tutorials/authentication_and_encryption/Understanding-Man-in-the-Middle-Attacks-ARP-Part1.html
3. **TELINOV, DIMITRI:** Atacuri SQL Injection, http://www.security.ase.md/publ/ro/pubro32/ ;
4. **WALKER, SHAUN & SANTRY, PATRICK J. Ş.A.:** Professional DotNetNuke ASP.NET"-2005.