CONTRIBUŢII LA CREŞTEREA DISPONIBILITĂŢII, SCALABILITĂŢII ŞI SECURITĂŢII SISTEMELOR DE COMUNICAŢIE

Teză destinată obţinerii
titlului ştiinţific de doctor inginer
la
Universitatea "Politehnica" din Timişoara
în domeniul ŞTIINŢA CALCULATOARELOR
de către

ing. Valer Bocan

Conducător ştiinţific: prof. dr. ing. Vladimir Creţu
Referenţi ştiinţifici: prof. dr. ing. Mircea Petrescu
prof. dr. ing. Victor Patriciu
prof. dr. ing. Mircea Vlăduţiu

Ziua susţinerii tezei: 12.06.2009

Seriile Teze de doctorat ale UPT sunt:

1. Automatică
2. Chimie
3. Energetică
4. Ingineria Chimică
5. Inginerie Civilă
6. Inginerie Electrică
7. Inginerie Electronică şi Telecomunicaţii
8. Inginerie Industrială
9. Inginerie Mecanică
10. Ştiinţa Calculatoarelor
11. Ştiinţa şi Ingineria Materialelor

Universitatea „Politehnica” din Timişoara a iniţiat seriile de mai sus în scopul diseminării expertizei, cunoştinţelor şi rezultatelor cercetărilor întreprinse în cadrul şcolii doctorale a universităţii. Seriile conţin, potrivit H.B.Ex.S Nr. 14 / 14.07.2006, tezele de doctorat susţinute în universitate începând cu 1 octombrie 2006.

România, 300159 Timişoara, Bd. Republicii 9
tel. 0256 403823, fax. 0256 403221
e-mail: editura@edipol.upt.ro

Cuvânt înainte

Scrierea acestui cuvânt înainte este la fel de anevoioasă ca și cercetarea din cadrul activității de doctorat, pentru că emoțiile și trăirile momentului strălucesc în comparație cu puterea de expresie a cuvintelor. Am să încerc totuși să aștern câteva rânduri de recunoștință celor ce au contribuit la formarea mea ca om, cerând iertare pentru stângăcia cu care o fac.

Mulțumesc domnului prof. dr. ing. Vladimir Crețu, conducătorul științific de doctorat, pentru ajutorul, sfaturile și îndrumarea continuă de pe parcursul lungului urcuș al activității mele de doctorat. De asemenea, recunoştinţa mea se îndreaptă către domnul prof. univ. dr. Victor Valeriu Patriciu de la Academia Tehnică Militară din Bucureşti pentru ideile şi sfaturile pertinente pe care le-am primit cu fiecare ocazie. Colegii mei din cadrul Alcatel-Lucent Romania mi-au fost de un nesperat ajutor în cercetare prin discuţii şi facilitarea accesului la documente pe care altfel cu greu le-aş fi putut găsi.

Deşi nu este specialistă în domeniu, ţin să mulţumesc soţiei mele Viorica pentru devotamentul ei, pentru răbdarea şi dragostea cu care m-a înconjurat în momentele grele. Pot spune că fără sprijinul nemijlocit al soţiei mele, această teză de doctorat nu ar fi existat. Mulţumesc şi părinţilor mei Vasile şi Dorina pentru anii grei şi lungi de-a lungul cărora m-au ocrotit şi mi-au oferit tot ajutorul de care am avut nevoie.

Mai presus decât tuturor, îi mulţumesc Bunului Dumnezeu pentru că mi-a ajutat să ajung până la acest hotar, că mi-a dăruit atâtea satisfacţii pe plan profesional şi personal şi pentru grija nemăsurată pe care mi-o poartă mie şi familiei mele.

Autorul,

Mai 2009

Fiicei mele Ioana Carina, a cărei naştere mi-a luminat existenţa…

Bocan, Valer

Contribuţii la creşterea disponibilităţii, scalabilităţii şi securităţii sistemelor de comunicaţie

Teze de doctorat ale UPT, Seria 10, Nr. 17, Editura Politehnica, 2009, 158 pagini, 39 figuri, 8 tabele.

ISSN: 1842-7707

ISBN: 978-973-625-881-7

Cuvinte cheie: comunicaţie, disponibilitate, scalabilitate, securitate, GSM, DoS, SSO, puzzle, distribuţie, conţinut

Rezumat: Teza de doctorat prezintă soluţii pentru ameliorarea disponibilităţii, scalabilităţii şi a securităţii sistemelor de comunicaţie. Sub acoperirea acestui deziderat integrator, se abordează trei obiective majore:

- Creşterea disponibilităţii şi scalabilităţii sistemelor de autentificare
- Studiul şi analiza problemei vulnerabilităţii reţelelor GSM la atacuri DoS şi elaborarea de propuneri pentru ameliorarea rezistenţei
- Propunerea unei arhitecturi de distribuţie digitală a conţinutului care îmbunătăţeşte într-o manieră semnificativă scalabilitatea şi maniera de cooperare între deţinătorii drepturilor de autor

Cuprins

Lista figurilor

1. Introducere

Societatea modernă a identificat o nouă necesitate vitală a sa: comunicaţia. Aceasta reprezintă forţa care animează comunităţile de indivizi sau chiar indivizii în sine, forţă care stă chiar la baza progresului umanitar. Felul în care omul a comunicat de-a lungul timpului s-a schimbat şi continuă să se schimbe pe măsură ce societatea evoluează. Dacă la început mijloacele de comunicare erau primitive, anevoioase, cu latenţă mare şi eficacitate generală redusă, astăzi schimbul de informaţie se produce cvasi-instantaneu pe tot cuprinsul globului. Comunicaţia este factorul determinant într-o multitudine de ramuri ale activităţii. Lipsa de coordonare pe un şantier, sincope în alimentarea cu subansamble a unei linii de producţie, imposibilitatea alertării asupra unui eveniment produs sau iminent care pune vieţi în pericol, reprezintă tot atâtea situaţii în care lipsa de comunicare duce la perturbarea ordinii fireşti sau mai rău, duce la pagube ce altfel ar putea fi prevenite şi evitate.

Scopul acţiunii de comunicare este de a transporta informaţia (cunoştinţele) între indivizii care compun societatea pe diverse căi şi modalităţi, într-o manieră care să asigure integritatea şi eventual confidenţialitatea comunicării. Comunicarea implică transmisia de informaţie între entităţi situate la diverse niveluri, iar eterogenitatea acestora (oameni – sisteme de calcul – linii de comunicaţie) implică ierarhizarea infrastructurii. Fiecare nivel al infrastructurii este capabil să înţeleagă doar acea informaţie care îi parvine într-o formă codificată potrivit cu nevoile sale. Spre exemplu, omul percepe uşor informaţie pe cale audio-vizuală, deci este nevoie ca nivelurile ierarhic inferioare să îi prezinte informaţia în această formă. Acest lucru se repetă până la nivelurile aflate la baza infrastructurii de comunicaţie.

Lipsa conjuncturală a comunicării face ca activitatea unor părţi din societate să fie perturbată, de la simple inconveniente legate de imposibilitatea efectuării unui apel telefonic până la consecinţe grave ca pierderi de vieţi omeneşti şi pagube materiale (luând ca exemplu nefuncţionarea unui sistem de avertizare la tsunami). Pentru ca situaţiile extreme să fie reduse cât mai mult posibil, sistemul de comunicaţie trebuie să fie de încredere atât din punctul de vedere al confidenţialităţii cât mai ales din punctul de vedere al disponibilităţii şi stabilităţii. Un sistem de comunicaţie care prezintă dese întreruperi în funcţionare sau variaţii largi ale calităţii serviciului va slăbi încrederea consumatorului, lucru ce va duce în cele din urmă la abandonarea utilizării lui, în detrimentul utilităţii sau confortului social.

1.1 Formularea problemei

Să luăm o formă de comunicare între indivizi, spre exemplu simplul act al vorbirii. Situaţiile în care vorbirea este împiedicată de o cauză oarecare sau

perturbată din cauza multitudinii de voci din apropiere sunt situaţii excepţionale cărora indivizii societăţii le găsesc mijloacele necesare pentru evitare sau corectare. În situaţiile în care comunicaţia este afectată, individul identifică cauzele şi ia măsurile de corecţie necesare, spre exemplu se deplasează într-o direcţie care să-i permită o comunicare bună cu interlocutorul. Se observă că părţile implicate în comunicaţie participă activ la îmbunătăţirea comunicaţiei prin acţiunile pe care le întreprind.

Cum societatea virtuală este o oglindire cvasi-fidelă a societăţii reale [75], situaţia este similară în cazul comunicaţiei electronice: imposibilitatea comunicării a două entităţi poate fi provocată de un atac de interzicere a accesului (Denial of Service – DoS), iar comunicarea cu o rată redusă poate fi dată de lipsa scalabilităţii sistemului de comunicaţie. Pentru ameliorarea acestor stări de fapt, sistemele implicate în comunicaţie trebuie să dispună de metode şi protocoale de acţiune pentru detectarea, eliminarea şi evitarea cauzelor care perturbă comunicaţia. Putem distinge trei parametri ai sistemelor de comunicaţie care definesc şi dau o măsură calităţii comunicaţiei:

- Disponibilitatea infrastructurii de comunicaţie – posibilitatea ca procesul de comunicaţie să aibă loc
- Scalabilitatea sistemelor de comunicaţie – capacitatea de a suporta volume crescute de trafic/încărcare fără degradarea severă a indicatorilor de performanţă
- Securitatea sistemelor de comunicaţie – capacitatea de a funcţiona în prezenţa unor factori perturbatori

Disponibilitatea unui sistem de comunicaţie reprezintă capacitatea acestuia de a fi gata a efectua o transmisie într-un timp rezonabil. Lipsa disponibilităţii se poate datora fie întreruperii fizice a căii de comunicaţie (fire telefonice, cablu de reţea, emiţător radio) sau se poate datora unui atac de tip Denial of Service (DoS) care împiedică transferul de date pe toată perioada sa. În prezent, rezistenţa la astfel de atacuri nu este un obiectiv de proiectare pentru majoritatea sistemelor de comunicaţie.

Scalabilitatea unui sistem de comunicaţii este capacitatea acestuia de a se adapta la diverse scenarii de încărcare. În cazul creşterii valorilor traficului, degradarea calităţii serviciului trebuie să se facă gradual şi proporţional, evitându-se astfel căderi rapide ale serviciului care ar putea fi exploatate ulterior de un atacator. În general protocoalele de distribuţie a conţinutului deservesc clienţii unul câte unul, lucru ce reprezintă o gâtuire a performanţei care s-ar putea obţine prin abordarea paralelă a distribuţiei.

Securitatea sistemului de comunicaţii reprezintă capacitatea sistemului de a funcţiona în condiţii normale sau apropiate de normal sub acţiunea unor factori externi perturbatori. În mod tradiţional când vorbim de securitate ne gândim la

protecţia şi confidenţialitatea datelor transmise de sistem, iar literatura de specialitate abundă de protocoale şi procedee de securizare a informaţiilor aflate în tranzit prin diverse medii. Disponibilitatea şi scalabilitatea sunt factori care afectează în mod direct nivelul de securitate al unui sistem de comunicaţii, de aceea considerăm că o abordare corectă a îmbunătăţirii securităţii presupune şi o creştere a celor doi factori.

1.2 Scopul

Scopul acestei lucrări de cercetare este de a prezenta soluţii pentru ameliorarea disponibilităţii, scalabilităţii şi a securităţii sistemelor de comunicaţie.

Reţelele GSM, ca parte a unui sistem larg de comunicaţii şi datorită răspândirii şi valorii ridicate de piaţă de care se bucură, ar putea constitui puncte vulnerabile în infrastructură. Atacurile asupra reţelelor GSM pot cauza pagube importante, de aceea acţiunile subversive sunt tentante pentru atacatori. Raţiunile atacurilor asupra reţelelor GSM par să nu difere de cele îndreptate împotriva reţelelor de calculatoare cum ar fi pierderi în venituri, impact social, etc. Securitatea în GSM se referă doar la partea radio şi este proiectată să împiedice doar atacatorii mai puţin experimentaţi. Deoarece resursele radio sunt limitate, sistemul GSM are facilităţi puternice de contabilizare a resurselor dar în cazul unor terminale neconforme cu standardul, resursele se pot aloca abuziv, neputându-se face nimic pentru constrângerea şi recuperarea acestora. Lucrarea de faţă ridică problema vulnerabilităţii reţelelor GSM la atacuri DoS şi face propuneri de ameliorare a rezistenţei.

În general, protocoalele de autentificare se bazează pe operaţii criptografice asimetrice şi pentru că acestea sunt operaţii complexe şi scumpe din punctul de vedere al resurselor, avem de-a face cu o vulnerabilitate la atacuri de tip Denial of Service (DoS) în cele mai multe cazuri. Dezechilibrul între resursele alocate de client pentru execuţia protocolului şi cele alocate de server permite clientului să angajeze în mod repetat resurse importante ale serverului. Acest lucru se traduce prin posibilitatea încărcării deliberate de către atacator a serverului de autentificare cu cereri false pentru care se alocă resurse de calcul, în detrimentul cererilor legitime care sunt deservite într-un timp mai îndelungat sau în cazul extrem, deloc. Lucrarea de faţă prezintă o posibilă soluţie la acest dezechilibru, şi anume creşterea artificială a costului de execuţie al protocolului pentru client, proporţional cu efortul cerut serverului. Astfel, în cazul autentificărilor sporadice încărcarea pe client este neglijabilă, dar dacă clientul emite cereri repetate către server cu scopul de a-l supraîncărca, încărcarea asupra sa creşte proporţional.

Multitudinea de dispozitive electronice portabile care ne înconjoară a atras implicit şi nevoia acestora de comunicare. De la simple mesaje scrise sau e-mail, aceste dispozitive sunt acum capabile de a reda conţinut digital audio şi video pe

care îl obţin fie de la o sursă locală cum ar fi un card de memorie fie de la o sursă online de tip streaming sau download. Consumatorii de entertainment digital mobil sunt în special tinerii, şi odată cu lansarea unui album de muzică nou sau al unui film, utilizatorii vor provoca creşteri momentane ale valorilor traficului care în lipsa unei infrastructuri suficient dimensionate pot constata degradări ale serviciului consumat, cum ar fi timpi de aşteptare mare, viteze de transfer scăzute, etc. Dimensionarea corespunzătoare a infrastructurii trebuie astfel dublată de eficientizarea transportului de informaţii. Pe piaţă există deja un număr de sisteme de distribuţie digitală a conţinutului, care se concentrează pe argumentele tehnice privitoare la mecanismul de redare a conţinutului pe dispozitivele autorizate. Scalabilitatea unor astfel de sisteme este redusă şi nu poate face faţă unor vârfuri de trafic şi în plus nici nu se are în vedere colaborarea între părţile care partajează drepturile asupra unui conţinut digital (autori). Lucrarea de faţă prezintă o arhitectură de distribuţie digitală a conţinutului care îmbunătăţeşte într-o manieră semnificativă scalabilitatea şi maniera de cooperare între deţinătorii de drepturi de autor.

1.3 Organizarea tezei

Capitolul 2 este o trecere în revistă a problematicii securității, disponibilității şi scalabilităţii sistemelor de comunicaţii, cu accent pe evidenţierea neajunsurilor ce survin datorită deficienţelor de design. Astfel, sistemele de autentificare, sistemele Single Sign-On şi reţelele GSM sunt vulnerabile la atacuri DoS, fiindu-le astfel afectată disponibilitatea. Vulnerabilitatea survine în urma alocării de resurse valoroase (putere de calcul, frecvenţe de comunicaţie, etc.) în condiţiile lipsei unui control strict. Deficienţele în design nu au ca urmare doar scăderea disponibilităţii, ci în cazul sistemelor de distribuţie a conţinutului pot afecta scalabilitatea. Datorită abordării secvenţiale a procesului, acest tip de sistemele întâmpină dificultăţi în manipularea unor cantităţi crescânde de date.

Capitolul 3 prezintă contribuțiile la creșterea disponibilității și scalabilității sistemelor de autentificare. Ideea principală este de a egala efortul depus de client cu cel depus de server pentru o operaţie de autentificare, lucru ce se realizează printr-un *client puzzle* (în esenţă inversarea parţială a unei dispersii). Pentru ca această soluţie să fie fezabilă în cazul clienţilor cu puteri de calcul care variază în limite largi (de la un simplu PDA sau SmartPhone până la reţele distribuite de calculatoare) s-au introdus conceptele de *threshold puzzle* respectiv *adaptive threshold puzzle.* Sistemele de autentificare de tip Single Sign-On prezintă particularităţi de comportament în cazul unui atac, exploatate cu ajutorul motorului de detecție *SSO-SENSE.* Capitolul prezintă fundamentarea matematică a acestor concepte precum şi aspecte practice de implementare şi măsurători de performanţă.

Capitolul 4 prezintă principalele vulnerabilităţi ale reţelelor GSM, clasificându-le după gravitate după o metrică folosită în software, ceea ce reprezintă

o premieră. Rezultatul deloc surprinzător este că cea mai gravă ameninţare asupra unei reţele GSM este atacul Denial of Service (DoS), vulnerabilitate similară sistemelor de autentificare dar pe partea radio.

Capitolul 5 propune o arhitectură de distribuţie a conţinutului menită să amelioreze problemele existente în prezent. Arhitectura prezentată abordează distribuţia prin prisma difuzării multiple (broadcast), utilizând un procedeu criptografic care să permită decodificarea conţinutului de către entităţi selectate. Deservirea mai multor clienţi simultan şi posibilitatea ca distribuţia să fie preluată de entităţi de tip proxy duce la creşterea scalabilităţii generale a sistemului.

Capitolul 6 prezintă concluziile tezei și prezintă perspectivele de cercetare în domeniu.

Publicaţiile personale sunt listate în anexa specială de la sfârşitul lucrării.

2. Securitate, disponibilitate şi scalabilitate în sisteme de comunicaţii

Pentru a fi utile scopului în care au fost proiectate şi construite, sistemele de comunicaţii trebuie să fie la dispoziţia utilizatorilor la momente oportune şi în condiţii bine stabilite, în plus trebuie să aibă un comportament previzibil în orice fel de condiţii de încărcare.

În comunicaţii, securitatea este o condiţie care rezultă din stabilirea şi menţinerea unor măsuri de protecţie care menţin starea de inviolabilitate faţă de acte ostile sau influenţe de acest fel, cum ar fi: impersonarea participanţilor, efectuarea de operaţii privilegiate, capturarea traficului în vederea analizei ulterioare, afectarea disponibilităţii sistemului prin acţiuni subversive, etc. Autentificarea, autorizarea şi criptarea sunt măsuri pentru controlul participanţilor şi sunt mecanisme interne sistemului de comunicaţie ca parte integrantă a acestuia.

Factorii care influenţează sistemele de comunicaţie sunt disponibilitatea şi scalabilitatea. Disponibilitatea este gradul în care un sistem sau subsistem este operabil şi într-o stare stabilă, la momentul începerii unei proceduri, când procedura este declanşată la un moment aleatoriu în timp. Scalabilitatea este factorul care influenţează în mod direct capacitatea de adaptare la încărcare. Spunem că un sistem de comunicaţie este scalabil atunci când poate deservi un trafic crescător într-o manieră în care degradarea performanţei se face liniar sau subliniar.

În cadrul acestui capitol se face o introducere în problema securităţii, disponibilităţii şi scalabilităţii sistemelor de comunicaţie cu accent pe probleme actuale.

2.1 Caracteristici ale sistemelor de comunicaţii

2.1.1 Securitatea

Conceptul de securitate în sistemele de comunicaţii se referă la o condiţie care rezultă din stabilirea şi menţinerea unor măsuri de protecţie care menţin starea de inviolabilitate faţă de acte ostile sau influenţe de acest fel. Starea de securitate se obţine prin stabilirea şi verificarea identităţii participanţilor de comunicaţie, autorizarea participanţilor şi protecţia datelor vehiculate.

Autentificarea este actul prin care se verifică identitatea pretinsă de un utilizator. O persoană care doreşte să efectueze o operaţie bancară trebuie să prezinte lucrătorului de la ghişeu o formă de identificare, uzual un act cu fotografie. Lucrătorul compară imaginea cu fizionomia persoanei, iar în caz de potrivire procedează la îndeplinirea cererii. În caz contrar, cererea este refuzată.

Autorizarea se referă la pasul ulterior autentificării, când odată stabilită identitatea, utilizatorul este verificat dacă are voie (are drepturi) să efectueze operaţia solicitată. Autorizarea se face pe baza unor liste de control acces.

În cele din urmă, operaţia (în cazul nostru comunicaţia) poate avea loc după un schimb de chei între participanţi din care să rezulte o cheie temporară de sesiune cu care comunicaţia va fi protejată.

2.1.2 Disponibilitatea

Gradul în care un sistem sau subsistem este operabil şi într-o stare stabilă, la momentul începerii unei proceduri, când procedura este declanşată la un moment aleatoriu în timp, se numeşte disponibilitate.

Disponibilitatea unui sistem de comunicaţii se poate defini ca:

$$A = \frac{T_{UpTime}}{T_{UpTime} + T_{DownTime}}$$

Unde T_{UpTime} şi $T_{DownTime}$ sunt duratele de timp aşteptate pentru funcţionarea, respectiv nefuncţionarea sistemului.

Nefuncţionarea sistemului de comunicaţii poate următoarele cauze:

- Nefuncţionarea dispozitivelor hardware implicate în comunicare
- Nefuncţionarea căii de comunicare
- Imposibilitatea realizării autentificării/autorizării/criptării, fapt ce duce la refuzul comunicaţiei

Deşi primele două cauze pot fi provocate şi deliberat, considerăm că defectele la nivelul hardware pot surveni în orice moment şi din cauze altele decât rezultatul unui atac. În ceea ce priveşte imposibilitatea autentificării, a autorizării sau a criptării într-un sistem de comunicaţie, aceasta poate surveni în urma unui atac de tip Denial of Service (DoS).

Atacurile de tip DoS sunt principala cauză de indisponibilitate a sistemelor şi de aceea este nevoie de măsuri de apărare cât mai eficace.

2.1.3 Scalabilitatea

Scalabilitatea este o noţiune în general dificil de definit. În general spunem că scalabilitatea este factorul care influenţează în mod direct capacitatea de adaptare la încărcare. Un sistem de comunicaţie scalabil deserveşte traficul crescător într-o manieră în care degradarea performanţei se face liniar sau subliniar. Un sistem scalabil poate să-şi mărească corespunzător capacitatea când beneficiază de o îmbunătăţire a infrastructurii hardware pe care rulează. Similar, un sistem

software este scalabil dacă poate suferi îmbunătăţiri de structură care să-i mărească capacitatea pe aceeaşi platformă hardware.

2.2 Securitatea şi disponibilitatea sistemelor de autentificare

Stabilirea identităţii participanţilor la o comunicaţie poartă numele de *autentificare* şi are ca rezultat autenticitatea, însemnând că partea care verifică (*verificatorul*) poate fi sigur că partea verificată (*verificatul)* este cel care pretinde că este. Uzual, tehnicile de autentificare se împart în trei categorii fundamentale:

- *Autentificare prin cunoştinţe* (ceva ce utilizatorul ştie: coduri PIN, coduri de tranzacţie, parole)
- *Autentificare prin posesie* (ceva ce utilizatorul are: chei, carduri de identificare sau alt fel de dispozitive fizice)
- *Autentificare prin proprietăţi* (identificarea biometrică a utilizatorului cum ar fi identificarea feţei, imagini ale retinei, şabloane vocale, amprente)

Având în vedere multitudinea de site-uri Internet, autentificarea preponderentă astăzi este cea prin cunoştinţe, urmată la mare distanță de cea prin posesie.

2.2.1 Autentificarea bazată pe parole

În cele mai multe sisteme distribuite şi reţele de calculatoare, protecţia resurselor se realizează prin login direct folosind parole, cu transmiterea în clar a acestora. Această autentificare are mai multe inconveniente, din care amintim doar câteva:

- Utilizatorii tind să selecteze parole neuniform distribuite. Această problemă este binecunoscută şi nu este neapărat legată de reţele de calculatoare şi sisteme distribuite. [35][50]
- Nu este convenabil pentru un utilizator care are mai multe conturi pe host-uri diferite să îşi amintească parola pentru fiecare, şi de asemenea să o introducă la fiecare schimbare a host-ului. În schimb, utilizatorul va alege să fie recunoscut de reţea ca întreg şi nu de host-urile individuale.
- Transmisia parolei este expusă la captură pasivă.

În principal datorită acestui ultim punct, autentificarea bazată pe parolă nu este potrivită în reţele de calculatoare şi sisteme distribuite. Parolele trimise prin reţea sunt foarte uşor compromise şi pot fi folosite ulterior pentru impersonarea utilizatorului.

În unele situaţii, autentificarea prin parolă este chiar deranjantă. Spre exemplu, la începutul telefoniei mobile în Statele Unite ale Americii, telefoanele mobile foloseau ca parolă internă la efectuarea unui apel chiar numărul telefonului

respectiv pentru ca centrala să poată factura fiecare apel în mod corect. Acest mod simplist de abordare a problemei a dus rapid la atacuri prin impersonare, rezultatul net fiind posibilitatea ca un atacator să efectueze convorbiri în contul altei persoane.

2.2.2 Autentificarea bazată pe adresă

O alternativă – nu neapărat mai sigură – la autentificarea prin parolă este autentificarea prin adresă. Aceasta presupune că identitatea unei surse se poate deduce din adresa acesteia conţinută în pachete. Ideea de bază este că fiecare host memorează identitatea celorlalte host-uri care au acces la resursele sale. În sistemul de operare UNIX, fiecare host are un fişier numit */etc/hosts.equiv*. Utilizatorii cu acelaşi cont pe ambele sisteme pot folosi utilitarele „r" fără a specifica vreo parolă.

Ideea de host-uri credibile nu este o soluţie la problema autentificării în reţele de calculatoare. De fapt, acest tip de autentificare chiar pune probleme mai mari din punct de vedere al securităţii. Dacă un atacator reuşeşte să intre pe contul unui utilizator dintr-un sistem, securitatea este compromisă pe toate sistemele care au încredere în acel sistem. În plus, administratorul de sistem nu poate da drepturi preferenţiale unor utilizatori.

În funcţie de mediul concret, autentificarea prin adresă este chiar mai puţin sigură decât cea bazată pe parolă. Avantajul îl constituie însă comoditatea în folosire şi de aceea unele sisteme au ales să o implementeze.

2.2.3 Autentificarea criptografică

Ideea din spatele autentificării criptografice este că A îşi dovedeşte identitatea către B prin efectuarea unei operaţii criptografice asupra unei entităţi cunoscute de ambii participanţi sau oferită de B. Operaţia criptografică efectuată de A se bazează pe o cheie criptografică. Aceasta poate fi fie o cheie secretă sau o cheie privată dintr-un sistem cu chei asimetrice.

În general, autentificarea criptografică este mai sigură decât autentificarea bazată pe parolă sau pe adresă. Tehnicile bazate pe dovezi *zero-knowledge* pot oferi mecanisme de autentificare puternice [101][70], însă acestea necesită calcule matematice destul de complexe şi prezintă mai multe facilităţi atractive pentru autentificare. Caracteristica definitorie este că părţile care se autentifică dovedesc că ştiu secretul fără a transfera efectiv informaţia către verificator.

În ciuda aparentei simplităţi, proiectarea sistemelor reale este foarte dificilă. O serie de protocoale publicate au prezentat erori de securitate substanţiale sau subtile. În timpul ultimei decade, eforturile de cercetare s-au concentrat în crearea de utilitare pentru dezvoltarea protocoalelor de autentificare şi distribuţie a cheilor cu o anumită asigurare formală a securităţii. Realizările cele mai notabile sunt logica

BAN [19] şi logica GNY [41]. În loc de a produce protocoale specifice, aceste metodologii se utilizează pentru a verifica un set de afirmaţii presupus adevărate.

O implementare cunoscută a autentificării criptografice este protocolul SSL, folosit în toate browserele populare cum ar fi Internet Explorer şi Firefox. Paragraful următor face o descriere a acestui protocol şi a proprietăţilor acestuia.

2.2.4 Protocolul SSL

Protocolul SSL este un protocol de securitate care oferă comunicare secretă prin Internet. Scopul principal al acestuia este de a oferi confidenţialitate şi încredere între două aplicaţii care comunică. Protocolul constă din două straturi: La cel mai de jos nivel, plasat deasupra unui protocol sigur de comunicaţie (spre exemplu TCP) se află protocolul *SSL Record*. Acesta este folosit pentru încapsularea protocoalelor de nivel mai înalt, cum ar fi protocolul *SSL Handshake* care permite serverului şi clientului să negocieze un algoritm de criptare şi cheile criptografice corespunzătoare înainte ca aplicaţiile să schimbe vreun mesaj. Un nivel superior al protocolului ar putea sta deasupra acestuia în mod transparent.

Protocolul SSL oferă securitatea conexiunii, având trei proprietăţi:

- Conexiunea este privată. Criptarea se utilizează după ce are loc schimbul iniţial pentru definirea cheii secrete. Criptografia simetrică se utilizează pentru codificarea datelor (DES, RC4, etc.)
- Identitatea părţilor se autentifică utilizând criptografie asimetrică (RSA, DSS, etc.)
- Conexiunea este de încredere. Transportul mesajului include verificarea acestuia cu un MAC parametrizat. Pentru calcularea MAC-ului se folosesc funcţiile de dispersie SHA, MD5, etc.

Scopurile protocolului SSL, în ordinea priorităţilor sunt:

1. **Securitatea criptografică**: SSL ar trebui să se utilizeze pentru conexiune sigură între două părţi.
2. **Interoperabilitate**: Programatori independenţi ar trebui să fie capabili de a dezvolta aplicaţii SLL care să funcţioneze cu succes fără ca aceştia să aibă cunoştinţă de codul scris de altcineva.
3. **Extensibilitatea**: SSL încearcă să furnizeze un cadru în care să se integreze metode noi de criptare (simetrică sau asimetrică), acest lucru contribuind la evitarea creării unui protocol nou (riscând implicit să se introducă noi slăbiciuni) şi evitarea scrierii unei biblioteci de securitate noi.
4. **Eficienţa relativă**: Operaţiile criptografice sunt scumpe din punctul de vedere al timpului, în special criptografia cu chei publice. Din acest motiv, SSL a înglobat un mecanism de caching al sesiunii pentru a reduce numărul de conexiuni ce trebuie stabilite în totalitate.

SSL este un protocol stratificat. Pe fiecare strat, mesajele pot conţine câmpuri pentru lungime, descriere şi conţinut. SSL primeşte mesajele de transmis,

fragmentează informaţia în blocuri prelucrabile, opţional comprimă datele, aplică un MAC, codifică şi în final transmite rezultatul. Datele recepţionate sunt decriptate, verificate, decomprimate şi reasamblate, apoi oferite protocolului superior ierarhic.

O sesiune SSL se poate afla într-una dintre stările predefinite ale protocolului. Este responsabilitatea protocolului SSL Handshake să coordoneze stările clientului şi ale serverului, permiţând ca sistemele să funcţioneze în mod consistent, în ciuda faptului că stările nu sunt exact paralele. În mod logic, starea este reprezentată de două ori, o dată ca starea curentă în operare şi (în timpul protocolului de handshake) ca starea în aşteptare. În plus, se menţin stări separate ale scrierii şi citirii. Când clientul sau serverul recepţionează un mesaj de schimb a specificaţiei cifrului, se copiază starea în aşteptare în starea curentă de citire. Când negocierea este completă, serverul şi clientul interschimbă mesajele de specificare a cifrului şi comunică prin noul cifru asupra căruia s-a convenit.

O sesiune SSL poate include mai multe conexiuni sigure, în plus participanţii pot avea mai multe sesiuni simultane. Starea sesiunii poate cuprinde următoarele elemente:

- **Identificator de sesiune**: O secvenţă arbitrară de octeţi aleasă de server pentru a identifica o sesiune activă sau reluabilă.
- **Certificat al egalului:** Certificat X.509 al părţii cu care se comunică. Acest element poate să fie nul.
- **Metoda de compresie:** Numele algoritmului utilizat pentru a comprima datele înainte de codificare.
- **Specificaţia cifrului:** Specifică algoritmul de codificare a datelor (cum ar fi nimic, DES, AES, etc.) şi un algoritm MAC (cum ar fi MD5 sau SHA). De asemenea defineşte atributele criptografice cum ar fi dimensiunea tabelei de dispersie.
- **Secretul master:** O valoare secretă de 48 de octeţi cunoscută de client şi de server.
- **Capacitatea de reluare:** Un indicator care arată dacă o sesiune se poate utiliza pentru a iniţia noi conexiuni.

Starea conexiunii poate include următoarele elemente:

- **Valori aleatoare pentru client şi server**: Secvenţe de octeţi alese de client şi de server pentru fiecare conexiune.
- **Secret MAC pentru scrierea pe server:** Secretul utilizat în operaţii MAC pe datele scrise de server.
- **Secret MAC pentru scrierea pe client:** Secretul utilizat în operaţii MAC pe datele scrise de client.
- **Cheia de scriere a clientului:** Cheia pentru datele codificate de server şi decodificate de client.
- **Cheia de scriere a serverului:** Cheia pentru datele codificate de client şi decodificate de server.
- **Vectori de iniţializare:** Când se utilizează un cifru bloc în modul CBC, se păstrează un vector de iniţializare pentru fiecare cheie. Acest câmp este iniţializat de protocolul SSL Handshake.

- **Numere de secvenţă:** Fiecare participant menţine numere de secvenţă pentru mesajele transmise şi recepţionate pentru fiecare conexiune. Când un participant trimite sau recepţionează un mesaj de modificare a specificaţiilor cifrului, acest număr se pune pe zero. Numere de secvenţă sunt de tipul *uint64* şi nu trebuie să depăşească valoarea $2^{64} - 1$ [37].

Sistemul este la fel de puternic ca cel mai slab algoritm de autentificare şi de distribuţie a cheilor. Astfel trebuie să se utilizeze doar funcţii criptografice sigure, verificate. Chei publice scurte, chei simetrice de 40 de biţi sau mai puţin şi servere anonime trebuie utilizate cu precauţie. Implementările şi utilizatorii trebuie să aleagă cu grijă autorităţile de certificare acceptabile, o autoritate mincinoasă putând provoca pagube imense.

2.2.5 Deficienţe ale protocoalelor de autentificare actuale

Deşi autentificarea criptografică oferă siguranţă participanţilor la comunicaţie, aceasta este susceptibilă la atacuri de tip DoS. În cele ce urmează se va face o scurtă prezentare a acestui tip de atac şi a cauzelor sale.

2.2.5.1 Atacuri DoS

Un atac DoS poate lua una din cele două forme posibile. Un atacator poate cauza netransmiterea de către reţea a mesajelor pe care ar trebui să le transmită în mod normal clienţilor săi. De cealaltă parte se află reţelele care trimit mesaje pe care nu ar trebui să le trimită. De departe cel mai cunoscut atac DOS este cauzarea de trafic fals (inundarea reţelei) în direcţia un server particular, lucru care în final va duce la împiedicarea clienţilor legitimi să obţină serviciul pe care îl cer de la acel server.

Comunitatea Internet cunoaşte o serie întreagă de atacuri DOS, cum ar fi atacurile prin inundare (flooding) şi atacurile prin pachete modificate.

O cauză a atacurilor este că dialogul iniţial între părţi are loc înaintea unei minime preautentificări. Serverul este incapabil de a distinge traficul legitim de cel fals, fapt ce are toate şansele să rămână aşa. Impunerea necesităţii de a autentifica toate cererile ar însemna un atac DOS în sine, din moment ce serverul ar fi ocupat cu verificarea semnăturilor digitale, indiferent dacă acestea sunt sau nu valide. Această nouă cale de atac ar fi la fel de periculoasă ca şi simpla umplere a tabelei TCP, în cazul unui atac de tip TCP SYN.

O altă cauză a atacurilor DoS este lipsa contabilizării resurselor. Spatscheck şi Peterson [80] consideră că sunt trei ingrediente cheie pentru protejarea de atacuri DOS:

[1] *contabilizarea* tuturor resurselor consumate de un client

[2] *detecţia* momentului când resursele alocate unui client depăşesc un prag prestabilit
[3] *constrângerea* – capacitatea de a revendica resursele blocate după detectarea unui atac prin alocarea de resurse minime unui atacator, lucru ce înseamnă automat evitarea unui atac DOS ulterior

În perioada când Internetul însuşi era proiectat, contabilizarea resurselor era scopul cu prioritatea cea mai mică, lucru ce ne afectează astăzi cel mai mult. În contrast cu reţelele de telefonie omniprezente unde folosirea resurselor este atent controlată, proiectanţii reţelei Internet nu au părut să acorde importanţă acestui aspect. Astfel, serverele alocă aceeaşi putere de calcul tuturor cererilor care sosesc la un moment dat ceea ce împiedică o degradare elegantă a performanţei în cazul unui atac sau în cazul unei încărcări excesive.

Scenariul anterior este oarecum similar cu mecanismul rudimentar de procesare a pachetelor de intrare datorită arhitecturii bazate pe întreruperi a subsistemului de reţea. Toate sistemele de operare implementează acest tip de arhitectură care s-a dovedit neadecvat în condiţii de încărcare mare. Pachetele de la intrare sunt procesate cu prioritatea maximă pentru ca apoi să fie distruse pentru că nu există nicio aplicaţie care să le deservească. Această situaţie se numeşte *receiver livelock*. Mai mult, chiar dacă există o aplicaţie care să deservească aceste pachete, prioritatea procesului nu este luată în calcul. Astfel, aplicaţiile cu prioritate mică primesc aceeaşi cantitate de trafic ca cele cu prioritate mai mare. În lucrarea lor, Druschel şi Banga propun o arhitectură cu procesare întârziată care se bazează pe demultiplexarea timpurie a pachetelor şi procesarea lor în funcţie de prioritatea destinatarului. Ei susţin că noua arhitectură ar îmbunătăţi stabilitatea, justeţea şi capacitatea sistemelor în condiţii de încărcare mare în timp ce performanţa nu ar avea de suferit în condiţii normale.

Crosby şi Wallach [27] au descris o nouă modalitate de atac bazată pe design-ul intrinsec al protocoalelor. Noua clasă de atacuri de bandă îngustă exploatează deficienţele structurilor de date în diferite aplicaţii. Structurile de date folosite în mod uzual au un timp de rulare în cazul mediu mult mai bun decât în cazul cel mai defavorabil. Spre exemplu, atât tabelele de dispersie cât şi arborii binari degenerează în liste înlănţuite atunci când la intrare se prezintă date alese corespunzător. Folosind banda tipică a unui modem, autorii citaţi au adus un server Bro în pragul colapsului; în şase minute de la începutul atacului, serverul ignora 71% din trafic şi consuma întreaga putere de calcul disponibilă.

2.2.5.2 Importanţa luptei împotriva atacurilor DoS

Având în vedere tendinţa globală a pieţelor de a se muta on-line, atacurile DoS se dovedesc mult mai periculoase decât s-a prevăzut la început întrucât acestea pot bloca victimele pe durate lungi de timp. De la momentul la care a început atacul şi până când acesta este detectat şi eliminat, victima este paralizată şi nu poate

răspunde la cererile legitime. Pentru site-urile comerciale mari aceasta se traduce prin pierderi financiare importante.

Deşi atacurile DOS nu ameninţă integritatea şi securitatea datelor în mod direct, nu există nici un motiv ca un altfel de atac să nu urmărească distrugerea şi alterarea lor. Atacurile pot viza date critice, cauzând astfel mai multe daune decât atacul DoS în sine.

Acest fel de atacuri în lanţ pot avea loc dacă protocoalele continuă dialogul cu atacatorul chiar şi după detectarea unor anomalii în dialogul purtat între părţi. Ideea de bază în spatele protocoalelor rezistente (numite protocoale *fail-stop* sau *fail-safe*) este ca schimbul de mesaje să înceteze cu orice client care nu urmează cursul firesc al protocolului.

2.3 Securitatea şi disponibilitatea sistemelor Single Sign-On (SSO)

În prezent, utilizatorii care doresc să acceseze servicii on-line trebuie să folosească un număr de credenţiale (perechi nume utilizator şi parolă) pentru a accesa fiecare serviciu la care sunt înregistraţi. Această fapt are neajunsuri din punctul de vedere al securităţii, deoarece utilizatorii se dovedesc incapabili a reţine un număr mare de parole de calitate, în plus costurile apelurilor la help desk (pentru resetarea parolelor) sunt importante tocmai din acest motiv.

Un sistem SSO abordează problema într-o manieră diferită. Utilizatorul se autentifică sistemului SSO cu credenţialele proprii, apoi orice acces ulterior la servicii este automat autentificat de către sistemul SSO, fără intervenţia manuală a utilizatorului. În prezent există mai multe arhitecturi SSO care vor fi prezentate în continuare, fiecare cu avantajele şi dezavantajele sale [67].

2.3.1 Tipuri de sisteme SSO

Sistemele SSO permit autentificarea automată a utilizatorilor către furnizorii de servicii. Autentificarea în sine implică identificare, deci un sistem SSO trebuie să aibă suport pentru managementul credenţialelor utilizatorului. Întrucât acestea pot lua diverse forme, acestea vor fi referite prin sintagma „identităţi SSO”.

Se pot deosebi două tipuri de sisteme SSO. Primul dintre acestea, numit „pseudo-SSO”, funcţionează ca un intermediar între utilizator şi aplicaţiile pe care acesta le accesează. Autentificarea iniţială (numită *autentificare primară*) se face între utilizator şi sistemul SSO. La fiecare acces la o aplicaţie care necesită credenţialele utilizatorului, sistemul SSO intervine şi efectuează paşii necesari pentru ca autentificarea să fie reuşită. Pentru fiecare acces la aplicaţie se efectuează o autentificare într-o modalitate transparentă pentru utilizator. Aplicaţia poată să nu fie informată că are loc o autentificare comandată de către sistemul SSO.

Întrucât credenţialele SSO sunt specifice fiecărui furnizor de servicii (aplicaţii software, etc.) relaţia identitate SSO/furnizor este *n : 1*, adică orice identitate dată corespunde unui singur furnizor.

Schema de principiu al unui sistem „pseudo-SSO” este prezentată în Fig. 1.

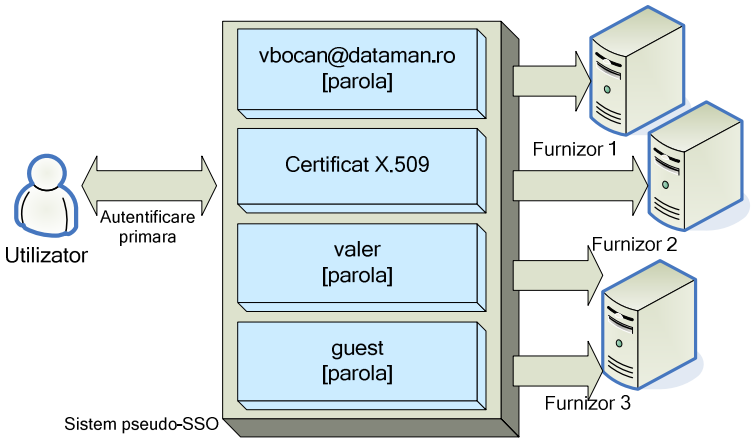


Fig. 1 Mecanismul de principiu al unui sistem „pseudo-SSO”

Un sistem „pseudo-SSO” este de fapt o aplicaţie care monitorizează comportamentul unor produse software care în mod tradiţional ar cere autentificarea utilizatorului în diverse moduri, cum ar fi ferestre de login, prompt, argumente în linia de comandă, etc. Prin mecanisme specifice, aceste cereri de autentificare sunt interceptate şi deservite în funcţie de credenţialele înregistrate pentru produsul în cauză, pentru utilizatorul autentificat curent. Din punct de vedere al mecanismelor de implementare, sistemele „pseudo-SSO” nu sunt altceva decât mecanisme de automatizare ale unor operaţii care în mod obişnuit ar fi efectuate de către utilizator. La fiecare acces la aplicaţie se face o autentificare în sine.

Dezavantajul acestui tip de sisteme SSO este manipularea în clar a credenţialelor din necesitatea ca acestea să fie livrate tot în clar furnizorilor de servicii. Protejarea credenţialelor în drumul lor de la sistemul SSO la aplicaţie este foarte dificilă, capturarea putându-se face prin diverse mijloace nu foarte avansate. Un alt dezavantaj este necesitatea actualizării scripturilor sau modulelor de interceptare ale ferestrelor de autentificare pentru fiecare produs monitorizat în parte. De regulă există un interval „mort” în care sistemul „pseudo-SSO” nu are cunoştinţe despre noile versiuni ale produselor, timp în care clienţii trebuie să aştepte. Pentru a-şi păstra clienţii, companiile care dezvoltă sisteme „pseudo-SSO” trebuie să fie în permanentă alertă, ceea ce duce la un consum crescut de resurse şi implicit la un cost ridicat.

Avantajul acestui tip de sisteme SSO este că pentru implementare sunt necesare schimbări minime în infrastructura software, întrucât aplicaţiile monitorizate nu trebuie să fie conştiente de noul sistem în funcţiune.

Cel de-al doilea tip de sisteme SSO este fundamental diferit de cel anterior şi va fi numit în continuare „true-SSO". În acest caz, autentificarea primară se realizează către o componentă numită *Authentication Service Provider (ASP)*. Uneori această componentă este numită *Identity Service Provider (ISP)*. Pentru a realiza SSO, ASP trebuie să aibă stabilită o relaţie cu fiecare furnizor de servicii (SP), relaţie care de regulă este stabilită printr-un contract. De asemenea trebuie să existe un canal securizat de comunicaţie între ASP şi SP.

Odată ce autentificarea primară a avut loc, la cererea furnizorilor de servicii (SP), ASP-ul realizează autentificarea prin intermediul unui protocol dedicat, folosind aşa-numitele *authentication assertions*. Acestea conţin identitatea utilizatorului şi starea sa aşa cum este cunoscută de către ASP şi trebuie transmise în mod securizat, în funcţie de specificul implementării. Spre deosebire de sistemele pseudo-SSO, relaţia identitate SSO / furnizor este *n : m*. Aceasta înseamnă că utilizatorul poate alege dintr-o plajă de identităţi pentru orice SP ales, dar aceeaşi identitate poate fi utilizată cu mai mulţi SP, dacă utilizatorul dispune acest lucru. Astfel este posibilă atribuirea de roluri specifice identităţilor SSO (care sunt de fapt pseudonime, aşa cum au fost definite în [68]).

Schema de principiu a unui sistem true-SSO este prezentată în Fig. 2.

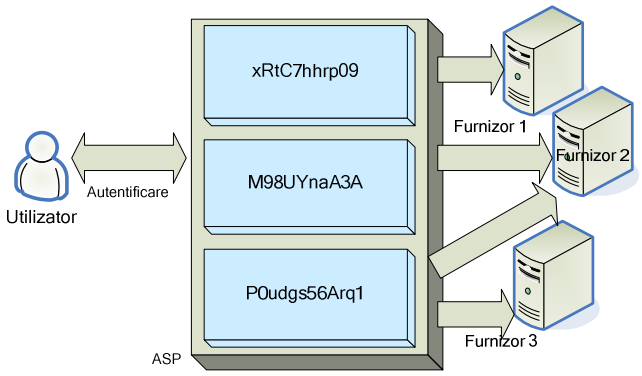


Fig. 2 Mecanismul de principiu al unui sistem „true-SSO"

ASP este un serviciu care deserveşte cereri provenite de la clienţi care au cunoştinţă de faptul că un sistem SSO este în funcţiune. Dezavantajul acestui sistem este necesitatea modificării produselor software. Acestea trebuie (re)proiectate şi implementate urmând specificaţiile sistemului SSO în cauză, lucru care nu este întotdeauna uşor pentru aplicaţii *legacy*.

Avantajul acestei abordări este că autentificarea se face o singură dată, credenţialele nu sunt stocate şi transmise în clar, putându-se totodată implementa anonimizarea, adică imposibilitatea creării de conexiuni între identităţile separate ale aceluiaşi utilizator.

Traseul informaţiei între părţile implicate este ilustrat în Fig. 3. În prima fază, utilizatorul efectuează autentificarea primară către ASP (1). Acest pas se efectuează o singură dată într-o sesiune de lucru. La un moment ulterior, utilizatorul doreşte să acceseze un serviciu oferit de un furnizor oarecare (2). Furnizorul trebuie să verifice la rândul său identitatea clientului, iar pentru acest lucru va solicita informaţiile de la ASP (3.1), care îi trimite un identificator ce reprezintă identitatea utilizatorului (3.2). În acest moment furnizorul de servicii poate trece la îndeplinirea iniţială a cererii utilizatorului (4).

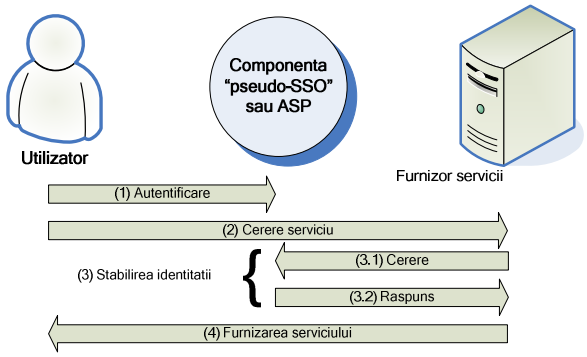


Fig. 3 Fluxul informaţional într-un sistem SSO generic

Modulele „pseudo-SSO"/ASP se pot afla pe calculatorul client (cel al utilizatorului) sau se poate afla pe o altă maşină pe care o numim „SSO proxy". Astfel, deosebim următoarele categorii de sisteme SSO [67]:

- Sisteme pseudo-SSO locale
- Sisteme pseudo-SSO bazate pe proxy
- Sisteme SSO locale
- Sisteme SSO bazate pe proxy

2.3.1.1 Sisteme pseudo-SSO locale

Într-un sistem pseudo-SSO local, componenta pseudo-SSO se află pe sistemul clientului. La începutul sesiunii de lucru, utilizatorul se autentifică componentei care tipic constă dintr-o bază de date care conţine identităţile utilizatorului pentru fiecare serviciu înregistrat în parte. O proprietate caracteristică este necesitatea de a păstra credenţialele în clar, de unde rezultă necesitatea ca

utilizatorul să aibă încredere în sistemul pe care lucrează. De remarcat că această arhitectură nu se pretează la sisteme publice sau unde accesul nu este restricţionat.

2.3.1.2 Sisteme pseudo-SSO bazate pe proxy

Într-un sistem pseudo-SSO local bazat pe proxy, componenta pseudo-SSO se află pe un sistem separat de cel al clientului. Autentificarea primară se efectuează la fel ca în cazul anterior la începutul sesiunii de lucru (eventual şi ulterior dacă proxy-ul doreşte reautentificarea periodică a clientului). Identităţile pentru fiecare furnizor de servicii în parte sunt stocate pe proxy, într-o bază de date securizată, însă tot este necesară prezentarea în clar a acestora în cazul dialogului cu un furnizor de servicii. Cererile de autentificare pot fi redirectate de către client către proxy printr-un mecanism special destinat acestui scop, fie sunt pur şi simplu interceptate fără ca programul în cauză să fie anunţat în vreun fel, astfel încât procesul să fie transparent pentru client. De remarcat că arhitectura se pretează la sistemele publice cu condiţia ca proxy-ul să fie securizat corespunzător.

2.3.1.3 Sisteme SSO locale

Un sistem SSO local are ca trăsătură principală furnizorul de identitate (ASP – authentication provider) plasat pe sistemul client. Acesta, cunoscând identitatea clientului dintr-o autentificare primară anterioară, va transmite furnizorilor de servicii identitatea clientului printr-un canal securizat. Între ASP şi furnizorii de servicii trebuie să existe o relaţie de încredere stabilită printr-un contract.

Diferenţa esenţială faţă de sistemele psuedo-SSO este că ASP-ul realizează autentificarea printr-un protocol special, iar credenţialele nu sunt transmise niciodată în clar.

2.3.1.4 Sisteme SSO bazate pe proxy

În acest model, rolul ASP-ului este luat de un server extern clientului, care acţionează ca un proxy între client şi furnizorii de servicii. Avantajul faţă de modelul anterior este separarea fizică între client şi ASP, în aşa fel încât acesta din urmă poate fi mai bine protejat împotriva încercărilor de fraudare. Este important de arătat că un ASP poate impersona oricare identitate pe care o reprezintă, rezultă deci că încrederea clientului şi a furnizorilor de servicii este esenţială pentru buna funcţionare a sistemului.

2.3.2 Proprietăţi ale sistemelor SSO

Sistemele SSO au o serie de proprietăţi trecute în revistă în cele ce urmează:

2.3.2.1 Confidenţialitatea

Confidenţialitatea are o importanţă deosebită în sistemele SSO [100]. Procesul de autentificare nu trebuie să dezvăluie identitatea directă a clientului. De asemenea, este de dorit ca două sau mai multe identităţi distincte ale aceluiaşi client să nu poată fi corelate fără consimţământul acestuia. Imposibilitatea corelării identităţilor implică imposibilitatea determinării identităţii personale a clientului pe baza credenţialelor sale [68].

Sistemele pseudo-SSO nu pot garanta pseudonimitatea (necorelarea identităţilor), deoarece acestea sunt specifice fiecărui furnizor de servicii şi pot conţine uneori date personale ale clientului. Astfel, unele sisteme pot cere autentificarea cu certificate, site-urile FTP cer uneori ca numele utilizatorului să fie chiar adresa e-mail, sau codul numeric personal.

2.3.2.2 Acces anonim la reţea

Deşi confidenţialitatea este o cerinţă a oricărei implementări SSO, aceasta depinde de modul particular în care este exploatat sistemul. Comunicarea între părţile implicate se realizează pe căi de comunicaţie „punct la punct", deci fiecare nod trebuie identificat în mod unic printr-o adresă. Din nefericire, comunicaţia pe straturile inferioare ale protocoalelor se face fără anonimizare, iar un eventual atacator poate compromite anonimitatea prin analiza pachetelor [4]. Fiecare pachet are în componenţă printre altele adresa sursă care poate oferi informaţii despre poziţionarea geografică a furnizorului de Internet. Cu această informaţie, atacatorul poate realiza corelarea identităţilor fără un efort semnificativ.

Problema determinării sursei traficului este rezolvată prin folosirea unor proxy-uri de anonimizare (ascundere a identităţii) prin care se realizează comunicaţia între client şi furnizorul de servicii. Proxy-ul are rolul de a înlocui adresa clientului cu propria sa adresă (posibil şi alte informaţii personale), făcând astfel imposibilă identificarea clientului. Nivelul de anonimitate este dat în cele din urmă de numărul de utilizatori care folosesc proxy-ul [68]. Exemple de astfel de servicii sunt Anonymizer [89] şi Freedom WebSecure [90].

Modelul de anonimizare cu un singur proxy nu protejează împotriva analizei de trafic şi se poate folosi atât timp cât operatorul proxy-ului nu este compromis. Din fericire există scheme – ca cele descrise în [24][40] – care protejează împotriva analizei traficului şi distribuie încrederea între proxy-urile implicate. O implementare de acest fel este JAP [91].

Anonimizarea poate fi realizată şi la nivelul proxy-ului SSO, într-o manieră transparentă pentru client şi pentru furnizorul de servicii. Desigur, întreg traficul clientului trebuie în acest caz rutat prin proxy, însă clientul nu va trebui să aibă încredere într-o altă entitate decât cea în care are deja încredere.

2.3.2.3 Mobilitatea utilizatorului

Sistemele SSO bazate pe proxy, prin însăşi structura lor permit ca utilizatorul să fie mobil, deoarece baza de date cu credenţialele sale sunt pe o maşină alta decât cea a clientului. Autentificarea se poate realiza de oriunde, în limitele impuse de protocolul de autentificare în sine.

Sistemele SSO locale oferă mobilitate utilizatorului în măsura în care baza de date cu credenţialele utilizatorului sunt păstrate extern clientului. Gradul de siguranţă este determinat în principal de modul de stocare al acestora, criptat sau în clar. Cele mai cunoscute implementări comerciale sunt Novell Secure Login (www.novell.com/products/securelogin), Passlogic V-Go (www.passlogic.com/sso), Protcom SecureLogin (www.protcom.cc), RSA ClearTrust (www.rsasecurity.com).

2.3.2.4 Utilizarea în medii ostile

În unele cazuri este necesar ca furnizorii de servicii să fie accesaţi din medii ostile, cum ar fi Internet cafe-urile şi terminalele publice. Este de dorit ca maşina clientului să nu aibă acces la secretele memorate în sistemul SSO, pentru a evita lansarea de atacuri prin impersonare (asumarea unei identităţi false). În acest caz este utilă implementarea modelelor cu proxy. Autentificarea primară trebuie în acest caz să se realizeze prin protocoale de autentificare adecvate, rezistente la atacuri replay sau man-in-the-middle.

2.3.2.5 Costuri de instalare și întreţinere

La capitolul costuri, instalarea şi întreţinerea sistemelor pseudo-SSO şi true-SSO sunt diferite. În cazul sistemelor pseudo-SSO, costurile de instalare sunt mici deoarece nu trebuie să existe o infrastructură de securitate propriu-zisă, iar furnizorii de servicii nu trebuie să aibă cunoştinţă de noul modul. Pe de altă parte, dacă furnizorul de servicii schimbă modul de autentificare al clientului, acest lucru trebuie să se reflecte în componenta pseudo-SSO, ceea ce duce la cheltuieli de exploatare. Lucrurile sunt cu atât mai complicate cu cât mediul în care sistemul pseudo-SSO este instalat este mai dinamic.

Situaţia este exact inversă în cazul sistemelor true-SSO. Cheltuielile legate de instalare sunt mari întrucât este necesară implementarea unei infrastructuri pentru ca dialogul client – furnizor de servicii să se realizeze în mod securizat. Acolo unde se impune este de asemenea necesară semnarea de contracte de servicii şi tratate de încredere reciprocă a părţilor implicate. Costurile de operare sunt în schimb reduse, deoarece schimbările părţilor nu implică automat modificarea interfeţei SSO.

2.3.2.6 Costuri de exploatare

Costurile de exploatare sunt mai mici în cazul sistemelor SSO locale faţă de cele bazate pe proxy, deoarece se induc costuri legate de operarea proxy-ului. Comunicaţia între părţile implicate poate fi taxată după un model negociat, mai ales când tot traficul clientului este rutat fizic prin proxy.

2.3.2.7 Relaţii de încredere

Atât în cazul pseudo-SSO cât şi în cazul true-SSO, clientul trebuie să aibă încredere în componenta care stochează credenţialele. În cazul sistemelor bazate pe proxy, relaţia de încredere se poate reglementa prin contracte şi politici la diferite niveluri pentru ca integritatea sistemului SSO să fie menţinută. În cazul sistemelor SSO locale, utilizatorul trebuie să aibă încredere în componenta software care stochează şi prelucrează credenţialele.

Dacă la sistemele true-SSO s-a putut evidenţia în mod clar posibilitatea stabilirii prin contract a relaţiei de încredere, în cazul pseudo-SSO relaţia de încredere este difuză, deoarece furnizorul de servicii poate să nu aibă cunoştinţă despre chiar existenţa sistemului SSO. Baza de date cu credenţiale poate fi stocată criptat sau în clar sau poate fi replicată pe mai multe servere, de unde rezultă că relaţia de încredere între client, furnizorul de servicii şi cel de identitate depinde de implementarea concretă a sistemului SSO. Mai mult, această relaţie este dinamică, în sensul că se poate schimba odată cu modificarea metodei de autentificare a furnizorului de servicii.

2.3.2.8 Rezolvarea conflictelor

În caz de conflict sau de cercetare legală, este necesar ca operatorul proxy-ului să poată furniza detalii despre tranzacţiile efectuate. Observaţia este valabilă atât în cazul sistemelor pseudo-SSO cât şi true-SSO, însă accentul cade pe sisteme true-SSO unde jurnalul trebuie să fie menţinut la zi cu toate tranzacţiile efectuate deoarece sunt implicate semnificativ mai multe entităţi decât în cazul pseudo-SSO.

2.3.2.9 Medii închise și medii deschise

Mediile închise în general nu au o politică de confidenţialitate bine conturată sau dacă există aceasta este considerată de importanţă redusă. În acest caz accentul se pune pe costurile de achiziţionare, exploatare şi întreţinere a sistemului SSO. Deoarece aceste costuri sunt mai mici în cazul sistemelor pseudo-SSO, sistemele scalabile la nivel de întreprindere sunt cele mai potrivite pentru recuperarea rapidă a investiţiei. Arhitectura acestor sisteme este analizată în [26].

În mediile deschise, nevoia de confidenţialitate este acută. Nivelul de confidenţialitate cerut este oferit doar de către sistemele true-SSO, astfel încât costurile legate de operare şi întreţinere sunt depăşite.

2.3.3 Deficienţe ale sistemelor SSO actuale

Implementările SSO curente cum ar fi Liberty [54][55][56], WS-Federation [88], Shibbolet [78] nu se pot impune ca sisteme complete de management al identităţii într-un mod pur tehnic. Literatura de specialitate menţionează necesitatea coroborării părţii tehnice a implementării cu partea juridică, sub forma unor înţelegeri şi relaţii de afaceri între părţile implicate (vezi modelul din Fig. 4)

Cu toate acestea însă, înţelegerile pot fi încălcate de participanţii la cercul de încredere dacă beneficiile sunt mai mari decât pierderile sau dacă unul sau mai mulţi furnizori de servicii sunt compromişi, caz în care participanţii oneşti au de suferit.

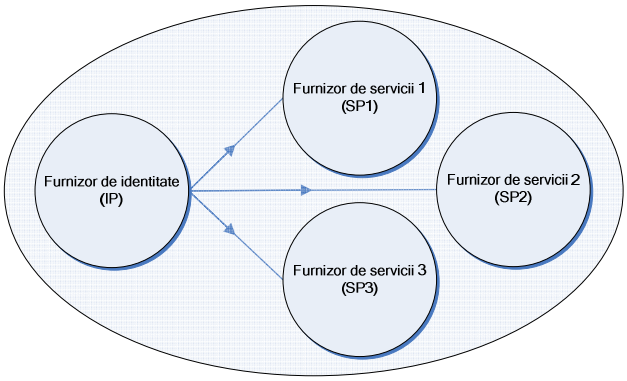


Fig. 4 Cercul de încredere bazat exclusiv pe relaţii juridice

Pe lângă aspectele legale, sistemele SSO au deficienţe la capitolul disponibilitate, fiind de cele mai multe ori vulnerabile la atacuri DoS. Ideea din spatele unui astfel de atac îndreptat împotriva unui sistem SSO cum este Liberty [54] este foarte de simplă (vezi Fig. 5).

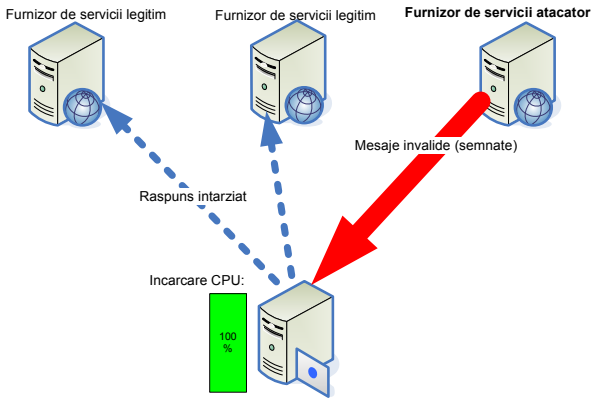


Fig. 5 Atacul DOS împotriva unui sistem SSO Liberty (indiferent de protocol)

Profitând de cerinţa ca mesajele între entităţi să fie semnate digital, un furnizor de servicii compromis poate inunda sistemul (în speţă furnizorul de identitate) cu mesaje care par legitime, cu semnătură falsă, trimise în cascadă. Nesuspectând nimic, furnizorul de identitate verifică semnăturile digitale ale tuturor mesajelor de intrare, lucru ce duce la epuizarea rapidă a resurselor, ceea ce se traduce fie prin răspunsuri întârziate la cererile furnizorilor legitimi de servicii (degradare de performanţă), fie prin absenţa cu desăvârşire a acestor răspunsuri (indisponibilitate).

Semnarea digitală a fiecărei cereri este o operaţie extrem de scumpă din punct de vedere al timpului de execuţie. Un server care îşi pune la dispoziţie resursele de calcul fără o minimă autentificare prealabilă este în pericol de a fi supraîncărcat cu cereri. În această idee, este foarte curios de aflat de ce dezvoltatorii din Liberty Alliance au trecut cu vederea acest aspect extrem de important. O posibilă explicaţie ar fi existenţa cadrului legal în care funcţionează sistemul, însă acest lucru nu este suficient pentru protejarea în fapt a participanţilor oneşti în sistem.

2.4 Securitatea şi disponibilitatea reţelelor GSM

În prezent, telefonia fără fir depăşeşte telefonia terestră ca număr de abonaţi în aproape toate ţările Europei şi Asiei. Terminalele mobile din generaţia GPRS/EDGE şi 3G facilitează accesul mobil la Internet, iar adoptarea pe scară largă a standardelor IEEE 802.11 şi Bluetooth permite integrarea platformelor de telefonie mobilă, laptop şi PDA cu suport pentru noi aplicaţii mobile puternice. Beneficiile imense ale reţelelor universale prezintă un set unic de riscuri care nu pot fi trecute cu vederea, având în vedere dependenţa din ce în ce mai mare de comunicaţie.

Tehnologia wireless, fie că e vorba de WiFi (802.11), Bluetooth, GSM sau 3G, este extrem de complexă. Din nefericire, inginerii de telecomunicaţii nu sunt experţi în domeniul securităţii şi prezintă în general tendinţa de a considera securitatea ca un produs auxiliar ce poate fi oricând adăugat produsului principal, dacă e nevoie. Acest mod de gândire este viciat, întrucât securitatea trebuie să fie în strânsă legătură cu tehnologia radio şi telecom. Aceste două domenii trebuie să coexiste şi să conlucreze pentru un scop comun, acela de a asigura integritatea, confidenţialitatea, autenticitatea şi disponibilitatea unui canal de comunicaţie.

O greşeală care se face în mod curent este că se consideră că procedurile de securitate existente sunt suficient de sofisticate încât să oprească atacurile de orice fel. Acest lucru este greşit, deoarece un atacator va ataca întotdeauna cea mai slabă verigă din sistem, iar acea verigă nu este aproape niciodată sistemul criptografic. Veriga cea mai slabă dintr-un sistem de comunicaţie fără fir este evident domeniul radio. O judecată de acest fel a dus la implementări cu grave carenţe de securitate în protocoalele 802.11b/g WEP şi Bluetooth [18]. Aceste sisteme nu au beneficiat de o analiză din punct de vedere al securităţii în nici una din fazele de proiectare, presupunându-se că mecanismele de securitate comerciale pot fi adăugate ulterior.

Securitatea în reţelele fără fir este o problemă importantă deoarece utilizatorii pot vehicula informaţii personale, importante sau critice printr-o infrastructură care nu este cu adevărat sigură. Slăbiciunile survin din folosirea multiplelor scheme de securitate incompatibile şi greşelile inerente din design-ul protocoalelor. Cel mai mare pericol este ca utilizatorul să perceapă întreaga structură ca fiind sigură, având încrederea că datele sale confidenţiale sunt în siguranţă. Mediul wireless prezintă multe probleme de securitate, cum ar fi confidenţialitatea, autentificarea părţilor, integritatea, autorizarea, non-negarea şi accesibilitatea. Alte probleme pot include uşurinţa în utilizare, viteza şi standardizarea [85].

Se poate afirma că strategia de securitate trebuie planificată şi implementată în funcţie de tipul datelor transportate şi de pierderile estimate în cazul în care are loc o scurgere de informaţie sau chiar datele sunt modificate în tranzit. De asemenea, trebuie să considerăm faptul că multe probleme de securitate apar datorită implementărilor greşite, interacţiunilor neplănuite între modulele unui sistem, creşteri neplanificate şi alte probleme de securitate apărute în urma atacurilor anterioare (vezi Fig. 6).

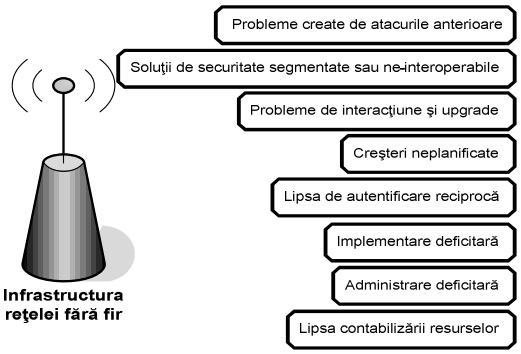


Fig. 6 Probleme de securitate ale reţelelor fără fir

Spre exemplu, un atac de tip Denial of Service, deşi în sine acesta nu corupe datele în mod direct, nu sunt motive să nu credem că o acţiune subversivă este în pregătire sau în desfăşurare [10].

Pentru a fi eficace cu adevărat, strategia de securitate trebuie aplică de la un capăt la celălalt, adică de la sursă la destinaţie, indiferent de calea pe care circulă informaţia. Spre exemplu, WAP oferă securitate prin WTSL (Wireless Transport Security Layer), însă această tehnologie nu este aplicată de la un capăt la celălalt întrucât criptarea are loc doar între terminalul mobil şi WAP Gateway [39].

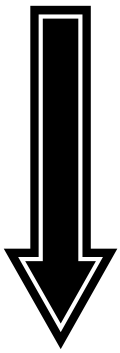
2.4.1 Mecanisme de securitate în reţelele GSM

Securitatea şi confidenţialitatea în GSM au fost principalele motive pentru care sistemul a fost considerat superior altor sisteme de comunicaţie mobile. Succesul deosebit a inspirat alte sisteme cum ar fi Code Division Multiple Access (CDMA), Personal Handy Phone System (PHS) şi Digital Enhanced Cordless Telecommunications (DECT). O inovaţie care a contribuit decisiv la popularitatea sistemului GSM a fost introducerea card-ului SIM (*Subscriber Identifier Module*) care a dus la separarea dispozitivului mobil de abonat. Card-ul SIM conţine *International Mobile Subscriber Identity* (IMSI) şi *Subscriber Identification Key* (K_i), ambele fiind folosite pentru autentificarea clientului în reţea. Securitatea GSM se bazează pe trei algoritmi: A3 şi A8 pentru autentificare şi A5 pentru criptare.

Cu mai mult de 1 miliard de abonaţi în lume, GSM este o ţintă potenţială pentru mai multe tipuri de atacuri. Cele mai uşor de pus în practică sunt atacurile care nu necesită tehnologie specială, cum ar fi redirecţionarea apelurilor către numere taxate speciale (în funcţie de operator), informaţii false la înregistrarea abonamentelor, fraudă în roaming şi furtul de terminale. Sistemele de monitorizare anti-fraudă monitorizează o varietate de indicatori cum ar fi apeluri multiple în acelaşi timp, variaţii largi ale valorilor facturilor, variaţii largi ale duratelor

convorbirilor (foarte scurte sau foarte lungi), schimbări în tiparele de folosire (indiciu că mobilul a fost furat sau este abuzat) şi monitorizarea îndeaproape a clientului în timpul unei perioade probatorii [38].

Tabel 1. Capacităţile unui atacator

Dificultate 	**Capturarea traficului** Reprezintă capacitatea unui intrus de a intercepta traficul şi informaţiile de semnalizare asociate utilizatorilor GSM. Echipamentul necesar este un telefon mobil modificat.
	Impersonarea unui utilizator Reprezintă capacitatea de a trimite date false şi/sau informaţii de semnalizare eronate către reţea cu intenţia de a părea că provin de la un utilizator legitim. Echipamentul necesar este un telefon mobil modificat.
	Impersonarea reţelei Reprezintă capacitatea de a trimite date false şi/sau informaţii de semnalizare către un utilizator cu intenţia de a le face să apară de la o reţea autentică. Echipamentul necesar este un BTS modificat.
	MITM – Man-In-The-Middle Reprezintă capacitatea unui atacator de a se plasa între reţea şi utilizatorul legitim cu scopul de a intercepta, modifica, şterge, reordona, reda şi falsifica data dintre cele două părţi. Echipamentul necesar este un BTS modificat în conjuncţie cu un telefon mobil modificat.
	Compromiterea Autentificării Reţelei Intrusul este în posesia unui vector de autentificare compromis (perechi challenge-response, chei de cifrare, chei de integritate, etc.)

Sistemul GSM prezintă mai multe probleme de securitate, după cum urmează:

- Comunicaţia şi traficul de semnalizare nu sunt protejate când se comunică cu reţelele fixe, deci reţelele GSM sunt doar atât de sigure cât reţelele fixe la care acestea se conectează.
- Infrastructura GSM nu adresează atacurile active, cum ar fi caching-ul identităţii, camparea pe un BTS fals, capturarea traficului, etc.
- Interceptarea legală a fost considerată ca o adăugire ulterioară.
- Mecanismele de autentificare şi criptografie sunt foarte dificil de îmbunătăţit.

- Lipsa transparenţei mecanismelor de securitate (utilizatorul nu este conştient cât de sigure sunt de fapt datele sale).

Capabilităţile tehnice ale unui atacator ce pot influenţa securitatea în reţelele GSM sunt în număr de cinci (vezi Tabel 1). Prima capabilitate este şi cea mai simplu de obţinut, următoarele implicând investiţii mai importante. Se consideră că un atacator care are o anumită capabilitate le are pe toate cele anterioare [38].

Capturarea traficului şi impersonarea utilizatorilor au fost două probleme cunoscute la momentul design-ului securităţii GSM.

2.4.1.1 Autentificarea

Autentificarea clientului se face printr-un algoritm cerere-răspuns simplu, după cum se arată în Fig. 7. Centrul de autentificare GSM (AuC) generează un număr aleator de 128 de biţi şi îl trimite staţiei mobile prin intermediul legăturii radio. Acest număr şi cheia abonatului (K_i) sunt introduse în algoritmul A3 care produce un răspuns semnat (SRES) care la rândul său este trimis înapoi la AuC. Între timp, AuC a calculat deja cantitatea SRES bazată pe aceleaşi date de intrare şi este acum în poziţia de a decide dacă staţia mobilă este cine spune că este. Sunt câteva probleme cu acest design. Algoritmii A3 (de autentificare) şi A8 (de generare a cheilor) sunt specifici fiecărui operator şi sunt secrete bine păzite. Acest lucru se traduce prin obscuritate mai degrabă decât securitate. Este binecunoscut faptul că un algoritm secret de autentificare sau de criptare poate fi vulnerabil deoarece nu beneficiază de experienţa comunităţii criptanalitice care ar putea să contribuie la descoperirea erorilor din design.

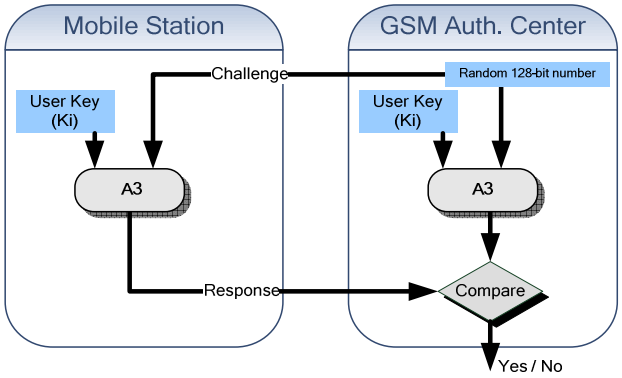


Fig. 7 Autentificarea în reţele GSM

În lumea software, când un program pretinde că implementează un nou algoritm sigur care este de câteva ori mai rapid decât DES sau AES, sunt şanse că

algoritmul nu este nimic mai mult decât o serie de XOR-uri. Cerinţa ca algoritmul să ruleze pe un smart-card (cum ar fi un SIM) are un profund impact asupra implementării practice. Astfel, 3GPP (3^{rd} Generation Partnership Project) sugerează ca implementările implicite ale A3 şi A8 să fie o simplă suită de operaţii XOR, fapt de demonstrează ideea noastră [1]. În mod surprinzător, faptul că SRES are doar 32 de biţi are un impact mic asupra securităţii în cazul unui atac de tip „zi de naştere"[1] deoarece această valoare este utilizată în conjuncţie cu cheia aleatoare de la AuC, numărul de capturi necesare pentru un atac cu o probabilitate fezabilă fiind astfel 1.84×10^{19} ($2^{128/2}$) faţă de 65536 ($2^{32/2}$). Mai multe informaţii despre atacuri de tip „zi de naştere" se găsesc în [81].

2.4.1.2 Criptarea

Spre deosebire de A3 şi A8, standardul GSM specifică algoritmul A5, folosit pentru criptarea vocii, datelor şi informaţiilor de semnalizare din reţeaua radio. Informaţia este codificată câte două cadre odată (2 x 114 biţi), câte unul pentru uplink şi downlink. În design-ul iniţial (numit A5/1), cheia de sesiune K este combinată cu un contor de cadre pentru a iniţializa un set de 3 registre care vor produce o ieşire de 228 de biţi prin aplicarea XOR la LFSR şi texul clar.

O implementare parţială a algoritmului GSM A5 a apărut pe Internet în iunie 1994. La acea vreme s-a susţinut faptul că aceea era doar un design timpuriu care are puţine asemănări cu algoritmul A5 folosit curent. Se spune că lungimea efectivă a cheii algoritmului A5 este de doar 40 de biţi [60].

Neînţelegerile dintre fabricanţii de telefoane celulare şi guvernul britanic cu privire la licenţele de export ale tehnologiilor de criptare în GSM s-au finalizat printr-un compromis în 1993. Naţiunilor vest-europene şi unor pieţe specializate cum ar fi Hong Kong li se permite accesul la tehnologia de criptare GSM A5/1, în timp ce o versiune mai slabă numită A5/2 a fost aprobată pentru export în celelalte ţări, incluzând naţiunile central şi est europene [58]. Aceasta este în principal o problemă politică ce implică drepturile individului şi capacitatea agenţiilor care asigură ordinea publică de a efectua supravegheri .

Design-ul algoritmului A5 (cu variantele sale A5/1, A5/2 sau mai recenta A5/3) a fost criptanalizat de Barkan, Bihan şi Keller [6] care au reuşit decodificarea

---

[1] Atacul de tip „zi de naştere" este un atac criptografic, numit astfel deoarece exploatează matematica din spatele paradoxului zilei de naştere din teoria probabilităţilor. Astfel, dacă o funcţie f(x) are rezultate egale probabilistic în mulţimea H (cu număr de elemente suficient de mare), se aşteaptă găsirea aleatoare a două soluţii x_1 şi x_2 cu $f(x_1) = f(x_2)$ după evaluarea a doar 1,25 x sqrt(H) argumente.

în timp real a fluxului GSM. Acest lucru pune într-o situaţie sensibilă pe oricine foloseşte infrastructura GSM pentru comunicaţii private.

Pentru uz casnic, securitatea în GSM se dovedeşte a fi net superioară celei din sistemele analogice. Folosirea autentificării, criptării şi a numerelor de identificare temporară asigură într-o anumită măsură confidenţialitatea şi anonimitatea utilizatorilor ca şi prevenirea utilizării frauduloase.

2.4.2 Deficienţe ale reţelelor GSM

Reţelele GSM, ca parte a unui sistem larg de comunicaţii şi datorită răspândirii şi valorii ridicate de piaţă de care se bucură, ar putea constitui puncte vulnerabile în infrastructură. Atacurile asupra reţelelor GSM pot cauza pagube importante, de aceea acţiunile subversive sunt tentante pentru atacatori. Raţiunile atacurilor asupra reţelelor GSM par să nu difere de cele îndreptate împotriva reţelelor de calculatoare, după cum urmează:

- Atacurile care cauzează întreruperea totală a serviciilor produc pierderi uriaşe în venituri, fără a pune la socoteală impactul social uriaş.
- Când nevoia de comunicare este imperioasă, spre exemplu după un atac terorist sau un dezastru natural, atacul DoS asupra reţelei GSM poate avea consecinţe dintre cele mai grave, cum ar fi pierderea de vieţi omeneşti şi pierderi materiale.

Atacurile care cauzează întreruperi parţiale sau intermitente ale serviciului GSM sunt extrem de greu de depistat. Un client care doreşte folosească reţeaua GSM poate avea dificultăţi în iniţierea apelurilor, încrederea în operatorul reţelei fiind erodată cu posibila consecinţă de migrare către concurenţă [14].

Deşi tehnologia GSM a fost proiectată având în vedere facilităţile avansate de securitate, acesta fiind şi motivul des invocat de superioritate asupra altor sisteme celulare, securitatea se referă doar la partea radio şi este proiectată să împiedice doar atacatorii mai puţin experimentaţi. Deoarece resursele radio sunt limitate, sistemul GSM are facilităţi puternice de contabilizare a resurselor. Totuşi, în cazul în care o staţie mobilă (telefon, modem, etc.) nu funcţionează conform standardelor, nu se poate face nimic pentru constrângerea şi recuperarea acestora, după cum se va vedea în cele ce urmează.

Scenariul tipic pentru partea preliminară a unui apel cu originea de la terminalul mobil (MS – Mobile Station) este următorul (vezi Fig. 8):

- MS-ul cere asignarea unui canal de control de la BSC (Base Station Controller).

- BTS (Base Station Transceiver) decodifică mesajul CHANNEL REQUEST, calculează avansul de timp (adică distanţa MS□BTS) şi trimite informaţia completă către BSC printr-un mesaj CHANNEL REQUIRED. De asemenea, se indică tipul de serviciu solicitat.
- După recepţionarea şi procesarea unui mesaj CHANNEL REQUIRED, BSC-ul informează BTS-ul despre ce tip de canal şi care număr de canal va fi rezervat în urma unui mesaj CHANNEL ACTIVE.
- BTS-ul confirmă primirea mesajului printr-un mesaj CHANNEL ACTIVE ACKNOWLEDGE.
- BSC-ul trimite către BTS mesajul IMMEDIATE ASSIGNMENT COMMAND, care la rândul său informează MS-ul de canalul alocat.

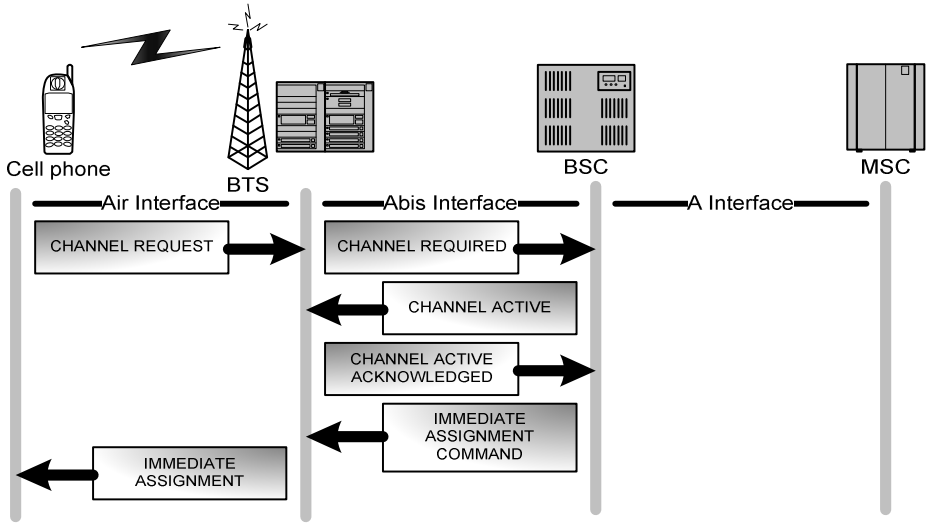


Fig. 8 Procesul de asignare a unui canal în GSM

În acest moment, la cererea unui terminal mobil **neidentificat** (client din perspectiva noastră), BSC-ul a alocat un canal de semnalizare din plaja de canale disponibile. Staţia mobilă este acum responsabilă de aderarea la restul protocolului, urmând a cere accesul la serviciu.

Întregul design se bazează pe faptul că staţia mobilă va urma în mod corect fiecare pas al protocolului. Ce se întâmplă însă daca o staţie mobilă repetă scenariul anterior şi cere mai multe canale de semnalizare fără ca vreodată să continue protocolul până la sfârşit? Deoarece numărul de canale de semnalizare este limitat, reţeaua se congestionează local iar cererile legitime sunt respinse datorită lipsei de canale disponibile. BSC-ul în cele din urmă va dezactiva cererile incomplete, eliberând resursele, dar acest lucru nu înseamnă recuperarea lor deoarece atacul în

sine nu este detectat. Canalele de trafic disponibile nu vor fi oferite clienţilor legitimi deoarece toate canalele de semnalizare sunt ocupate (vezi Fig. 9).

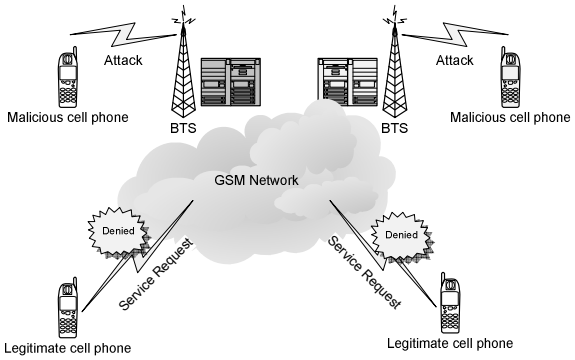


Fig. 9 Atacul de tip DoS într-o reţea GSM

Chiar dacă reţeaua GSM ar efectua o autentificare minimă a staţiei mobile cerând acesteia numărul IMEI sau nivelul de putere recepţionat de la cele şase celule învecinate, atacul tot ar fi posibil deoarece aceasta are un control complet asupra informaţiilor, putând raporta valori false.

Este interesant de remarcat că întărirea în sine a securităţii SIM-ului nu este relevantă, întrucât atacul este asupra protocolului de iniţiere a apelului în sine şi nu asupra SIM-ului [14].

2.5 Securitatea şi scalabilitatea sistemelor DRM

Sistemele DRM (Digital Rights Management) au ca scop principal aplicarea unor tehnici pentru restricţia liberei circulaţii a conţinutului digital (în orice formă, dar mai ales video şi audio) în scopul protejării proprietăţii intelectuale a autorului.

Punctul de vedere al susţinătorilor sistemelor DRM este că fără un sistem de protecţie al conţinutului, care să permită unui grup restrâns de clienţi accesul la conţinutul digital, rata pirateriei ar creşte într-o măsură mare care în final duce la diminuarea profiturilor autorilor. Odată cu declinul vânzărilor, susţinătorii DRM spun că factorii creativitate şi apoi calitate vor avea de asemenea de suferit.

Susţinătorii libertăţilor civile susţin că folosirea tehnicilor DRM ar trebui limitate sau chiar îndepărtate, întrucât extinderea controlului autorilor de conţinut digital până după faza vânzării dăunează expresiei creative şi drepturilor consumatorilor. Orice conţinut digital ar trebui să fie protejat de legea drepturilor de autor care însă să permită utilizatorului onest folosirea liberă a conţinutului în anumite situaţii, cum ar fi vizionarea sau audiţia privată acasă pe dispozitive

diferite, copierea de siguranţă, efectuarea de copii pentru membrii familiei, etc. Sistemele DRM actuale nu sunt capabile să facă concesii pentru utilizarea liberă în anumite situaţii, multe voci afirmând că prin protejarea conţinutului digital se aduce atingere drepturilor legale ale consumatorilor.

Printre primele sisteme DRM şi poate cel mai contestat este CSS (Content Scrambling System) [96], folosit pentru protejarea filmelor pe suport DVD. Sistemul a fost dezvoltat de DVD Consortium ca un instrument de influenţare a producătorilor de hardware de a produce numai sisteme fără anumite caracteristici. Accesul la cheia de decriptare a CSS a fost permis doar producătorilor care au fost de acord să nu includă caracteristici ca digital-out, care ar fi permis ca filmul să poată fi copiat uşor într-o formă fără pierderi. În acest fel DVD Consortium este în cele din urmă capabil de a dicta politica hardware pentru întreaga industrie DVD.

La scurt timp după implementarea sistemului CSS, algoritmul acestuia a fost spart. Au apărut programe cum ar fi DeCSS care permit copierea filmelor şi vizionarea copiilor pe dispozitive hardware care în mod normal nu ar fi capabile sau pe sisteme de operare altele decât cele pentru care player-ele software au fost făcute iniţial disponibile. În Statele Unite ale Americii şi în alte ţări ale lumii au apărut legi cum ar fi Digital Millennium Copyright Act (DCMA) care în mod explicit interzic folosirea programelor sau a altor dispozitive de ocolire a protecţiilor DRM.

Deşi sistemele DRM sunt cel mai adesea folosite pentru protejarea conţinutului video acestea încep să fie folosite şi pentru alte tipuri de conţinut. Fişierele audio cumpărate din multe magazine online au diverse tipuri de scheme DRM ataşate cu scopul de a limita numărul de dispozitive care le pot reda. Mulţi producători de publicaţii electronice folosesc implementări DRM similare pentru a limita numărul de computere pe care acestea se pot vizualiza şi chiar numărul de vizualizări.

Problemele de securitate, problemele legate de utilizarea legală şi expresia creativă sunt pe primul front al disputei din jurul sistemelor DRM. Această dispută va continua fără îndoială în anii care urmează, susţinută de industria media care consideră că DRM este singura modalitate prin care îşi pot salva modelul de business. Totuşi, un număr de inovatori în domeniu au început să exploreze alternative, anticipând înfrângerea sistemelor DRM. Un exemplu este firma Apple condusă de Steve Jobs care a anunţat că piesele muzicale vor fi oferite în magazinul online propriu atât în format protejat cu DRM cât şi în format liber.

2.5.1 Deficienţe ale sistemelor DRM

Motivul existenţei sistemelor DRM este protejarea conţinutului digital şi acordarea permisiunilor de redare a acestuia unui subset de clienţi autorizaţi. Scopul final al oricărui atac îndreptat împotriva DRM este obţinerea conţinutului în clar

pentru a fi ulterior distribuit liber şi pentru ca redarea pe diverse dispozitive neautorizate să fie posibilă.

Sistemele DRM pot fi atacate în trei moduri: atacuri asupra infrastructurii, atacuri asupra unităţii securizate de stocare şi atacuri asupra modulului de redare [83].

2.5.1.1 Atacuri asupra infrastructurii

Atacurile de acest tip nu au ca scop principal recuperarea cheilor de conţinut sau accesarea conţinutului în sine, ci dezactivarea completă a funcţiei DRM sau perturbarea bunei sale funcţionări, în sensul împiedicării furnizării serviciului pentru unul sau mai multe dispozitive (clienţi).

Dezactivarea funcţiei DRM poate avea loc cel mai probabil în cazul unui dispozitiv mobil, aceasta putând surveni ca urmare a unui atac produs fără consimţământul sau ştirea proprietarului, prin modificarea neautorizată a softului. Acest atac poate avea ca scop subminarea încrederii utilizatorului în sistemul DRM care protejează un anumit conţinut media, ceea ce duce în final la renunţarea completă la sistem şi scăderea cotei sale de piaţă.

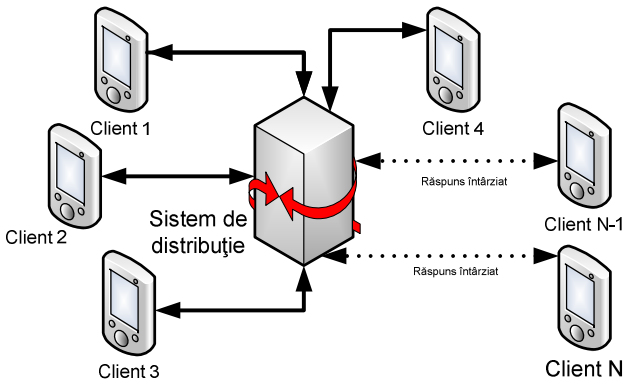


Fig. 10 Scalabilitate redusă într-un sistem de distribuţie al conţinutului, cu DRM.

Atacul prin împiedicarea furnizării de conţinut digital protejat cu DRM poate avea loc la nivel de infrastructură şi se traduce în fapt printr-un atac DoS. Acţiunea de protejare cu DRM a unui conţinut digital este în sine o operaţie costisitoare din punct de vedere al resurselor, aceasta presupunând din partea serverului una sau mai multe operaţii criptografice cu chei publice, operaţii cu chei simetrice şi aplicarea funcţiilor de dispersie asupra conţinutului. În cazul în care sistemul de distribuţie este de tip centralizat, deservind o mulţime de clienţi, acesta devine un

punct de gâtuire datorită slabei proprietăţi de scalare. Astfel, clienţii care solicită un conţinut digital de la server vor primi răspunsul cu întârzieri proporţionale cu gradul de încărcare al sistemului (vezi Fig. 10), încărcare care depinde în mod direct de numărul simultan de solicitări.

2.5.1.2 Atacuri asupra unităţii securizate de stocare

Deoarece orice sistem DRM are în componenţă un modul criptografic care decodifică conţinutul protejat cu ajutorul unor chei memorate în cadrul dispozitivului sau într-un loc ascuns din cadrul sistemului de operare, anumite atacuri pot viza exact această zonă. În funcţie de design, atacul poate viza:

- Obţinerea cheilor asociate dispozitivului – clonarea dispozitivului mobil vizat se poate face uşor iar conţinutul destinat dispozitivului original poate fi redat pe toate clonele. Acest atac are însă un efect temporar, întrucât anumite scheme DRM au un sistem de revocare a cheilor compromise care fac neutilizabile clonele.
- Obţinerea cheilor asociate conţinutului digital – acest lucru se traduce prin posibilitatea de decodifica conţinutul digital şi de a îndepărta protecţia DRM. Conţinutul astfel obţinut este liber de orice restricţii şi poate fi redat pe orice dispozitiv, cu sau fără DRM.

Cea mai cunoscută metodă de stocare securizată este TPM – Trusted Platform Module [92], un dispozitiv criptografic ce se doreşte a fi implementat în orice PC nou. Caracteristica cheie ce permite TPM să ofere stocare securizată este capacitatea de a stoca securizat o serie de măsurători. Pentru a realiza acest lucru, TPM are un set de registre numite PCR – Platform Configuration Registers. Un PC echipat cu TPM şi un BIOS securizat poate efectua o serie de măsurători ale hardware-ului şi software-ului de pe maşina în cauză şi să le memoreze în registrele menţionate. TPM-ul poate semna criptografic aceste registre şi le poate trimite unui terţ, spre verificare. Acesta la rândul său, poate verifica măsurătorile şi poate afirma că sistemul a pornit în configuraţia menţionată de TPM. Mai mult, TPM nu permite interogarea registrelor de către o maşină a cărei configuraţie hardware sau software a fost modificată.

2.5.1.3 Atacuri asupra modulului de redare

Acest tip de atac vizează ultima legătură din lanţul DRM şi pentru că atacatorul nu are nevoie de cunoştinţe de criptografie, acest tip de atac este şi cel mai uşor de pus în practică. În principiu atacul constă în interceptarea semnalului digital în clar după ce acesta a fost autorizat şi decriptat de către sistemul DRM. Dacă interceptarea se poate face înainte de transformarea semnalului în formă analogică (pentru redare pe TV sau în difuzoare), calitatea semnalului capturat este bună, de multe ori identică cu originalul, ceea ce se traduce într-un atac reuşit. Dacă însă interceptarea are loc după transformarea în semnal analogic, captura se

realizează cu pierderi de calitate, atacul având astfel un succes parţial (Fig. 11 Atac asupra modulului de redare dintr-un sistem DRM).

Pentru a împiedica interceptarea conţinutului digital în tranzit de la mediul de stocare la dispozitivul de redare, compania Microsoft a patentat şi implementat în sistemul de operare Windows Vista un mecanism numit Protected Media Path (PMP), care este în esenţă un set de tehnologii DRM care creează un aşa-numit mediu protejat (PE – Protected Environment). Ideea din spatele acestui concept este că aplicaţiile de nivel înalt (ex. player-ele video şi audio) operează la un nivel înalt de abstractizare, putând da comenzi simple gen „Start”, „Stop”, etc., fără a avea acces la date în sine. Mediul protejat (PE-ul) este disponibil doar aplicaţiilor agreate şi semnate de Microsoft, alte aplicaţii neavând acces [97].

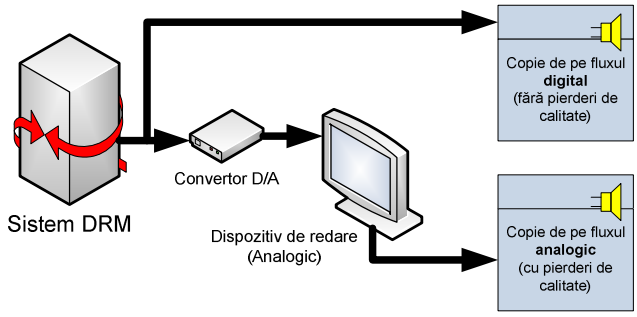


Fig. 11 Atac asupra modulului de redare dintr-un sistem DRM

2.6 Concluzii

Sistemele de comunicaţii sunt indispensabile în societatea modernă. Un simplu acces la Internet, un apel de voce sau de date prin telefonia mobilă sau ascultarea muzicii la banalul gadget de buzunar, sunt acţiuni care necesită căi de comunicaţie. Lipsa sau întreruperea comunicaţiei poate avea urmări dintre cele mai diverse şi de cele mai multe ori serioase.

Securizarea căilor de comunicaţie este un domeniu în care contribuţiile sunt variate. Unul dintre domeniile în care încă există carenţe în abordare este creşterea disponibilităţii şi asigurarea scalabilităţii sistemelor de comunicaţie, această nişă fiind acoperită de prezenta lucrare de cercetare.

Deşi eterogene din punct de vedere tehnologic, s-a avut în vedere un orizont larg de sisteme, cum sunt:

- Sistemele de autentificare
- Sistemele Single Sign-On (SSO)
- Reţelele GSM
- Sistemele de distribuţie a conţinutului digital

Capitolul de faţă a făcut o descriere a fiecărui sistem amintit şi a scos în evidenţă deficienţele existente actual, după cum urmează:

- Sistemele de autentificare, sistemele Single Sign-On (SSO) şi reţelele GSM sunt vulnerabile la atacuri de tip DoS. Vulnerabilitatea apare din cauza lipsei de control asupra resurselor alocate participanţilor la comunicaţie şi are de cele mai multe ori ca rezultat degradarea severă a serviciului sau blocarea lui pe perioada desfăşurării atacului.
- Sistemele DRM actuale sunt puţin scalabile şi indirect sunt susceptibile la atacuri DoS.

Capitolele următoare vor prezenta pe larg contribuţiile aduse în domeniu, ca soluţii propuse pentru fiecare dintre deficienţele descrise aici.

3. Contribuţii la creşterea disponibilităţii şi scalabilităţii sistemelor de autentificare

Interacţiunea sistemelor de diverse tipuri a dus automat la necesitatea stabilirii cu o oarecare precizie a identităţii, proces numit uzual autentificare. Procesul de autentificare a fost îndelung analizat în numeroase lucrări, cele mai multe punând în mod firesc accentul pe latura securităţii.

Cu toate acestea, un nivel de securitate ridicat al protocolului nu reprezintă implicit o condiţie suficientă pentru ca autentificarea să se soldeze cu succes. Este nevoie ca autentificarea să aibă loc la modul concret, adică părţile implicate în autentificare să fie disponibile şi să execute procesul într-un timp rezonabil.

În general protocoalele de autentificare se bazează pe operaţii criptografice cu chei publice şi pentru că acestea sunt operaţii complexe şi scumpe din punctul de vedere al resurselor, avem de-a face cu o vulnerabilitate la atacuri de tip Denial of Service (DoS). Dezechilibrul între resursele alocate de client pentru execuţia protocolului şi cele alocate de server permite clientului să angajeze în mod repetat resurse importante ale serverului. Acest lucru se traduce prin posibilitatea încărcării deliberate de către atacator a serverului de autentificare cu cereri false pentru care se alocă resurse de calcul, în detrimentul cererilor legitime care sunt deservite într-un timp mai îndelungat sau în cazul extrem, deloc. Soluţia la acest dezechilibru îl reprezintă creşterea artificială a costului de execuţie al protocolului pentru client, proporţional cu efortul cerut serverului. Astfel, în cazul autentificărilor sporadice încărcarea pe client este neglijabilă, dar dacă clientul emite cereri repetate către server cu scopul de a-l supraîncărca, încărcarea asupra sa va creşte proporţional.

Metoda numită Client Puzzles a fost propusă în literatură pentru echilibrarea efortului pentru cele două părţi implicate în autentificare [84]. În acest capitol se propune adoptarea şi îmbunătăţirea Client Puzzles în cadrul sistemelor de autentificare clasice şi a celor de tip Single Sign-On, care să ducă la ameliorarea efectelor atacurilor DoS.

3.1 Client Puzzles

Ideea de bază a acestui mecanism de protecţie împotriva atacurilor DoS este că un client care doreşte să participe la execuţia protocolului de autentificare să-şi aloce resursele în mod proporţional cu cele ale serverului, iar în orice punct al execuţiei protocolului de autentificare, costul rulării pentru client să fie mai mare decât pentru server. Costul pentru client poate fi crescut artificial prin solicitarea rezolvării unui puzzle al cărui grad de dificultate poate fi stabilit cu uşurinţă de către

server. În acelaşi timp, verificarea corectitudinii soluţiei nu trebuie să fie o povară pentru server deoarece acest lucru ar anula beneficiile acestei tehnici.

Într-o lucrare din anul 1978, Merkle [62] a fost primul care a venit cu ideea de puzzle-uri criptografice dar a aplicat ideea doar pentru schimbul de chei şi nu în autentificare. Client puzzle-urile au fost aplicate în atacurile TCP SYN de către Juels şi Brainard [48] care menţionează că SSL are aceeaşi slăbiciune şi dau o demonstraţie riguroasă a caracteristicilor de securitate. Aura, Nikander şi Leiwo au aplicat puzzle-urile în protocoalele de autentificare în general [84] iar Dwork şi Naor [33] au propus măsuri de reglementare pentru mesajele nesolicitate. Rivest, Shamir şi Wagner [72] discută o problema conexă a criptografiei blocată în timp.

3.1.1 Descriere

Înainte să aloce resurse pentru deservirea unei conexiuni serverul trebuie să se asigure că pe întreaga durată a execuţiei protocolului costul pentru client este mai mare decât pentru server. Costul pentru client se poate mări artificial prin solicitarea rezolvării unui puzzle care trebuie să aibă anumite proprietăţi, după cum urmează [84][102]:

- Crearea unui puzzle şi verificarea soluţiei nu necesită resurse importante din partea serverului.
- Costul rezolvării puzzle-ului este uşor de a fi modificat de la 0 la infinit.
- Puzzle-ul poate fi rezolvat pe majoritatea platformelor hardware.
- Precalcularea soluţiei puzzle-ului este imposibilă.
- În timp ce clientul rezolvă puzzle-ul, serverul nu trebuie să memoreze soluţia sau alte informaţii specifice clientului.
- Acelaşi puzzle poate fi distribuit mai multor clienţi. Cunoscând soluţiile calculate de unul sau mai mulţi clienţi nu ajută în calcularea unei noi soluţii.
- Un client poate reutiliza un puzzle prin crearea mai multor instanţe ale sale.

Principiul general este arătat Fig. 12.

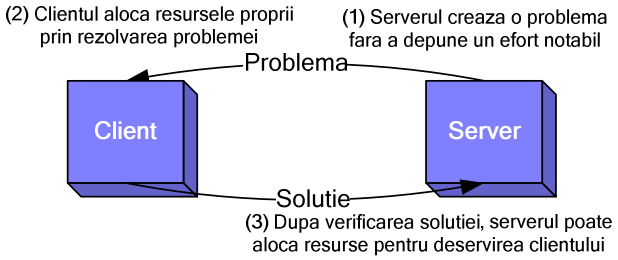


Fig. 12 Principiul client puzzles

3.1.2 Tipuri existente de puzzle-uri

3.1.2.1 Inversiunea parţială a unei funcţii de dispersie

Prima soluţie menţionată în literatură este inversiunea parţială a unei funcţii de dispersie [48][84]. Ideea de bază este identificarea unei cantităţi căreia aplicându-i-se o funcţie de dispersie, să producă un rezultat cu o configuraţie prestabilită, spre exemplu un număr de biţi într-o configuraţie convenită.

Funcţiile de dispersie sunt primitive criptografice simple care necesită efort de implementare redus pe o varietate de platforme. Deşi inversarea parţială a funcţiilor de dispersie este o operaţie paralelizabilă, propunerile ulterioare de puzzle-uri ale unor autori prezintă la rândul lor neajunsuri. Inversiunea parţială rămâne astfel alegerea din cadrul lucrării de faţă ca bază de plecare pentru cercetare.

3.1.2.2 Inversiunea logaritmului discret

În [86] se propune folosirea mecanismului Diffie-Hellman pentru a construi un puzzle, având ca bază inversiunea logaritmului discret şi se introduce noţiunea de bastion. În accepţiunea lucrării, bastionul este o entitate securizată care protejează un număr de servere.

Ideea de bază este următoarea: serverul şi bastionul se pun de acord asupra unui grup ciclic finit G şi unui element generator g, care se presupune cunoscut de către toţi atacatorii. Serverul alege un număr natural a şi trimite g^a bastionului şi similar, bastionul alege un număr natural b şi trimite g^b serverului. Atât serverul cât şi bastionul au acum o cantitate $(g^a)^b = (g^b)^a = g^{ab}$ care poate fi transmisă parţial clientului, spre exemplu cu un număr l de biţi lipsă din b. Clientul trebuie să determine valoarea lui b prin inversarea logaritmului discret.

3.1.2.3 Puzzle-uri blocate în timp

În [72] s-a propus folosirea unui puzzle blocat în timp, care funcţionează ca o capsulă de timp care constă dintr-un mesaj ce poate fi descifrat doar după trecerea unui timp.

Se presupune că se doreşte criptarea unui mesaj M pentru o perioadă de T secunde. Mesajul M se criptează cu un algoritm cunoscut folosind o cheie K (spre exemplu RC5), C_M = RC5(K,M). Cheia K se criptează la rândul ei cu $C_K = K + (a^2)^t \pmod n$, cu t = T * S, unde S este numărul de ridicări la putere pe secundă pe care le poate efectua cel care rezolvă puzzle-ul. În lipsa factorizării numărului n, recuperearea cheii K se face cel mai eficient efectuând t operaţii.

3.1.2.4 Puzzle-uri înlănţuite

Conceptul de puzzle-uri înlănţuite a fost introdus de Ma [59] în 2005 şi de Groza [44] un an mai târziu. Construcţia puzzle-ului prezentat în [59] începe cu alegerea unui număr aleatoriu iniţial h_0, asupra căruia serverul aplică în mod repetat o funcţie de dispersie pentru a obţine lanţul h_0, h_1, …, h_k unde $h_{i+1} = \text{hash}(h_i)$ iar k este lungimea dorită a lanţului. Acest calcul aduce un avantaj serverului prin stocarea întregului lanţ, verificarea soluţiei reducându-se la o simplă căutare într-o tabelă.

Schema propusă în [44] este similară ca tehnică cu licitaţiile puzzle propuse de Wang şi Reiter [87], adică cu cât clientul calculează mai multe verigi ale lanţului, cu atât mai multe resurse i se vor aloca de către server.

Deşi cele două scheme rezolvă problema calculului paralel a soluţiei puzzle-ului, acestea prezintă la rândul lor neajunsuri serioase: schema propusă de Ma necesită memorarea soluţiei fiecărei verigi a lanţului – ceea ce poate duce la epuizarea resurselor de stocare ale serverului – iar cea propusă de Groza este vulnerabilă la atacuri de tip zi de naştere pentru anumite configuraţii ale lanţului. În plus, această schemă necesită trei operaţii hash per verigă pentru producerea unui lanţ, un cost relativ ridicat în condiţii de atac.

3.1.3 Crearea unui puzzle

Periodic (spre exemplu o dată la câteva minute), serverul generează o valoare aleatoare $\mathbf{N_S}$. Pentru a împiedica atacurile prin ghicire, valoarea trebuie să aibă o entropie de 64 de biţi şi să nu fie o valoare predictibilă în vreun fel, ca de exemplu amprenta de timp. Această entropie ar trebui să fie suficientă pentru a împiedica un atacator să calculeze perechi «N_S, rezultat». Potrivirile ocazionale rezultate din atacuri de tip „zi de naştere” nu au efecte prea importante în acest caz.

Serverul trebuie să decidă de asemenea şi nivelul de dificultate **k** al puzzle-ului pe baza unui cumul de măsurători ale condiţiilor curente de încărcare. În rezumat, puzzle-ul trimis clienţilor arată astfel:

$$« N_S, k »$$

- N_S – valoarea aleatoare generată de server (de obicei o cantitate cu mărimea de 64 biţi)
- k – nivelul de dificultate a puzzle-ului

3.1.4 Rezolvarea puzzle-ului

Pentru a rezolva puzzle-ul, clientul generează la rândul său o valoare aleatoare $\mathbf{N_C}$. Scopul acestei valori este dublu. În primul rând, dacă clientul

refoloseşte valoarea $\mathbf{N_S}$ generată de server, poate construi un nou puzzle prin generarea unei noi valori N_C. În al doilea rând, fără această valoare, un atacator ar putea calcula puzzle-ul înaintea clientului şi ar trimite rezultatul serverului. 24 de biţi de entropie ar trebui să fie îndeajuns pentru a împiedica atacatorul de a epuiza întreaga plajă de valori ale N_c dat fiind faptul că N_S se schimbă frecvent.

Clientul trebuie să aplice în mod repetat o funcţie de dispersie unei cantităţi iar puzzle-ul se consideră rezolvat când primii k biţi ai cantităţii Y sunt egali cu 0.

$$\mathbf{h(C, N_S, N_C, X) = Y}$$

a) h – funcţie criptografică de dispersie, cum ar fi MD5 or SHA
b) C – identitatea clientului
c) N_S – valoarea aleatoare generată de server
d) N_C – valoarea aleatoare generată de client
e) X – soluţia puzzle-ului

Întrucât serverul schimbă N_S periodic, atât timp cât acest N_S se consideră recent, serverul trebuie să menţină o listă de perechi N_S – N_C în aşa fel soluţiile vechi să nu poată fi refolosite.

Deoarece nu se cunoaşte nicio metodă ocolitoare pentru a determina X, singura posibilitate este ca această cantitate să se determine secvenţial, adică prin forţă brută. Nivelul de dificultate k (adică numărul de biţi 0 de la începutul lui Y) dictează cât de mult va lua rezolvarea puzzle-ului. Dacă k este egal cu 0 nu este necesar nici un efort, iar dacă k este egal cu 128 (pentru MD5) sau 192 (pentru SHA), clientul trebuie să inverseze o întreagă funcţie de dispersie, lucru cvasi-imposibil.

3.1.5 Dificultatea puzzle-ului

Parametrul k reprezintă dificultatea puzzle-ului. Sarcina de a stabili această valoare este destul de dificilă deoarece nu există nicio metrică ce s-ar putea folosi într-o implementare reală. În conformitate cu [28], cea mai potrivită abordare ar fi luarea în calcul a numărului de operaţii RSA alocate deja. Încărcarea curentă a procesorului şi numărul de conexiuni la intrare sunt metrici care depind de un număr de factori care le fac nepotrivite în implementările reale.

Din nefericire, timpul de rezolvare în funcţie de dificultatea puzzle-ului urmează o curbă exponenţială, aplicabilitatea practică fiind astfel limitată. Pentru a rezolva un puzzle de dificultate k, clientul trebuie să efectueze în medie aproximativ 2^{k-1} operaţii. În [84], Aura, Nikkander şi Leiwo susţin că valorile rezonabile pentru k sunt între 0 şi 64. Prin experiment am aflat însă că plaja rezonabilă este însă mult mai îngustă.

Pentru a obţine o scală mai precisă pentru parametrul de dificultate al puzzle-ului, Juels şi Brainard [48] au propus împărţirea problemei în puzzle-uri mai

mici de dificultate egală care ar urmau să fie rezolvate separat iar rezultatul total să fie obţinut din combinarea rezultatelor parţiale. Aura, Nikkander şi Leiwo [84] susţin că aceeaşi granularitate poate fi obţinută din combinarea sub-problemelor de dificultate diferită dar la un cost mai redus pentru server, fără a demonstra practic soluţia.

3.1.6 Neajunsuri ale soluţiei Client Puzzle

Client Puzzles este o alegere potrivită securizarea protocoalelor de autentificare împotriva atacurilor DoS atât timp cât atacul nu este distribuit [74]. Deoarece dificultatea puzzle-ului este stabilită după o metrică ce ia în calcul doar angajamentul serverului, ignorând puterea de calcul a clientului (care poate varia în limite largi), eficacitatea metodei poate să fie limitată în anumite cazuri. Puzzle-urile sunt generate la momente precise, momente la care se ia în calcul angajamentul instantaneu al serverului (numărul de operaţii criptografice care urmează să fie executate într-o perioadă de timp imediat următoare sau orice altă metrică), dar acest lucru poate să nu fie potrivit în toate cazurile.

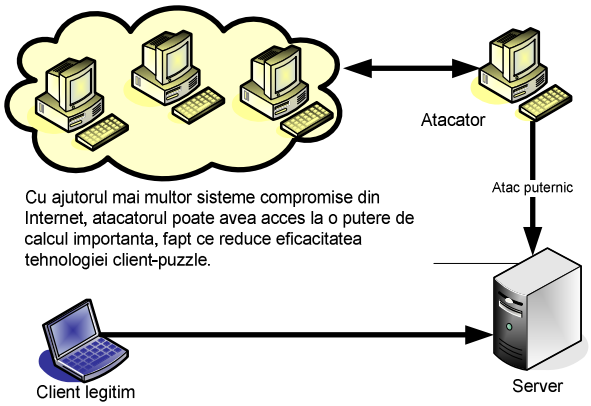


Fig. 13 Schema unui atac puternic

Puzzle-urile sunt astfel vulnerabile la o formă particulară de atac, numită de aici înainte **atac puternic**, datorită naturii paralelizabile a puzzle-ului. Un atac puternic se defineşte ca un atac de tip *denial-of-service* pus la cale de un atacator cu acces la o putere de calcul importantă. În acest caz, atacatorul este capabil de a rezolva puzzle-urile într-un timp mult mai scurt decât un client legitim, după cum se poate vedea în Fig. 13.

Să presupunem că un server autentifică un număr iniţial de clienţi legitimi, dificultatea iniţială a puzzle-ului fiind zero. Procesul de autentificare îşi urmează cursul normal, până când are loc un atac puternic. În acest moment, serverul are

tendinţa de a creşte gradual nivelul de dificultate spre valori superioare, pentru a ţine pasul cu cantitatea de procesare necesară pentru deservirea cererilor atacatorului. Deşi nivelul de dificultate poate fi crescut până la infinit, acest lucru înseamnă un atac DOS în sine îndreptat asupra clienţilor legitimi care nu ajung niciodată să rezolve un puzzle atât de dificil.

Deşi improbabil şi greu de pus în practică, un atac puternic este posibil. Dacă un atacator ar avea acces la *N* sisteme de calcul conectate (cu *N* suficient de mare ca să vorbim de putere de calcul semnificativă), atunci timpul necesar rezolvării puzzle-ului de dificultate *k* ar fi proporţional mai mic. Programul SETI [77] şi efortul de a sparge algoritmul RSA [32] sunt exemple reale de cum pot conlucra sute de mii de sisteme pentru acelaşi scop comun. Puterea cumulată a reţelei *distributed.net* a depăşit echivalentul a 160000 de calculatoare PII / 266MHz, lucru ce arată faptul că atacurile puternice sunt posibile.

3.1.7 Propuneri pentru ameliorarea experienţei clientului

Pentru că în forma actuală experienţa utilizatorului în relaţia cu un sistem de autentificare protejat cu Client Puzzles este suboptimă, propun o serie de modificări ale design-ului actual reunite sub denumirea Threshold Puzzles [11].

3.1.7.1 Limitarea superioară a nivelului de dificultate

Design-ul curent al tehnologiei client puzzle [84] specifică o plajă a dificultăţii între 0 (nici un efort necesar) şi 128 sau 192 (teoretic imposibil, în funcţie de tipul de funcţie de dispersie folosită). Datorită faptului că dificultatea are o creştere exponenţială, implementările acestui mecanism sunt limitate la a folosi valori ale dificultăţii dintr-o plajă mult mai restrânsă. Nivelurile ridicate ale dificultăţii ar putea conduce la atacuri DoS îndreptate asupra clienţilor legitimi, deoarece clienţii ar petrece un timp apreciabil pentru găsirea soluţiei.

Pentru ca percepţia dificultăţii puzzle-ului să fie optimă pentru client trebuie ca timpul de rezolvare al acestuia să fie inferior timpului în care serverul poate răspunde unei cereri de la client având în vedere încărcarea curentă, adică:

$T_{client} <= T_{server}$

Definim timpul de rezolvare al puzzle-ului pentru client ca fiind $T_{client} = M * t_c$, unde M este numărul mediu de operaţii necesare pentru rezolvarea puzzle-ului iar t_c este timpul mediu per operaţie din punctul de vedere al clientului.

Ştim că $M = \frac{\sum_{i=1}^{2^k} i}{2^k} = \frac{2^k(2^k+1)}{2^{k+1}} \sim 2^{k-1}$, deci $T_{client} = 2^{k-1} * t_c$.

Timpul în care serverul poate răspunde unei cereri ia în calcul dimensiunea curentă a cozii de aşteptare Q precum şi viteza sa t_s, astfel: $T_{server} = Q * t_s$.

Inegalitatea devine:

$2^{k-1} * t_c <= Q * t_s$, de unde obţinem valoarea maximă a dificultăţii k:

$k <= \log_2 (Q * t_s / t_c) + 1$

Deoarece k trebuie să fie o cantitate pozitivă, pentru a cuprinde şi cazul în care încărcarea pe server este redusă (cantitatea de sub logaritm ar fi subunitară în acest caz), avem:

$k <= \log_2 (\max(1, Q * t_s / t_c)) + 1$, unde:

Ec. 1

- Q – dimensiunea curentă a cozii de aşteptare
- t_s – timpul necesar efectuării unei operaţii criptografice, pentru server
- t_c – timpul necesar efectuării unei operaţii criptografice, pentru client

Pentru t_s = 0,003 şi t_c = 0,5 graficul comparativ între dificultatea proporţională a unui client puzzle şi dificultatea limitată a unui threshold puzzle arată ca în Fig. 14. Se observă că în cazul threshold puzzle percepţia degradării performanţei pentru client este mai puţin pronunţată.

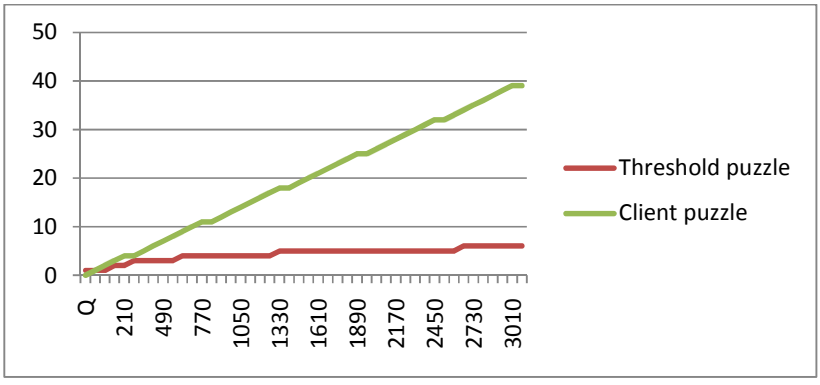


Fig. 14 Evoluţia comparativă a dificultăţii puzzle-urilor TP şi CP

3.1.7.2 Stabilirea unui timp minim de răspuns

Pentru ca un atac să aibă succes, este necesar ca atacatorul să fie capabil de a trimite cereri numeroase serverului într-un timp scurt, în pofida instalării

tehnologiei client puzzles. Pornind de la observaţia simplă că orice puzzle, indiferent de nivelul său de dificultate, are un timp de rezolvare, serverul poate să refuze cererile ale căror puzzle a fost rezolvat într-un timp prea scurt.

Ideea de bază constă în adăugarea unei amprente de timp la care serverul a generat valoarea sa aleatoare la lista «N_S, N_C, X, k». Când serverul primeşte soluţia, acesta poate calcula timpul exact de care a avut nevoie clientul pentru a rezolva puzzle-ul. Acest timp nu ar trebui să fie mai mic de o estimare a serverului bazată pe gradul de dificultate. Dacă este, atunci rezultă că serverul se află sub un atac puternic şi ar trebui să înceteze imediat comunicarea cu clientul în cauză.

După cum s-a arătat în paragraful anterior, în medie pentru rezolvarea unui puzzle de dificultate k este nevoie de 2^{k-1} operaţii, deci timpul estimativ de rezolvare este:

$$T_e = (2^{k-1}) * T_{operaţie}$$

- T_e – timpul estimat pentru rezolvarea puzzle-ului
- k – nivelul de dificultate stabilit de server
- $T_{operaţie}$ – timpul mediu de efectuare a unei operaţiuni criptografice

3.1.7.3 Algoritmul Threshold Puzzles

Algoritmul Threshold Puzzles complet este următorul:

1. La începerea comunicaţiei cu un client, serverul verifică starea sistemului după o metrică proprie. Dacă încărcarea nu depăşeşte un prag considerat critic, execuţia algoritmului se opreşte, deoarece sistemul de apărare nu este necesar.
2. Serverul generează o valoare aleatoare şi unică $\mathbf{N_s}$ (nonce) cu o entropie de 64 de biţi şi memorează momentul de timp $\mathbf{T_S}$ la care generarea a avut loc.
3. Serverul stabileşte un nivel de dificultate **k** pentru puzzle şi estimează timpul minim $\mathbf{T_e}$ de rezolvare al acestuia. Valoarea dificultăţii se limitează la un prag superior stabilit, pentru ca o creştere necontrolată a acestuia să nu aibă repercusiuni asupra clienţilor cu putere de calcul limitată.
4. Serverul creează un puzzle pentru client:

$$« N_S, k, T_S »$$

   - N_S – valoarea aleatoare generată de server
   - k – nivelul de dificultate al puzzle-ului, cu $k <= \log_2 (\max(1, Q * t_s / t_c)) + 1$, unde:
     a. Q – dimensiunea curentă a cozii de aşteptare
     b. t_s – timpul necesar efectuării unei operaţii criptografice, pentru server
     c. t_c – timpul necesar efectuării unei operaţii criptografice, pentru client

- T_S – timpul la care valoarea aleatoare a fost generată

5. La recepţia puzzle-ului, clientul verifică faptul că amprenta de timp T_S este recentă, generează un număr aleator N_C, şi rezolvă puzzle-ul aplicând în mod repetat o funcţie de dispersie unei cantităţi formate din C, N_S, N_C şi X. Puzzle-ul se consideră rezolvat când k biţi consecutivi ai cantităţii Y sunt egali cu 0.

$$h(C, N_S, N_C, X) = Y$$

   a) h – funcţie criptografică de dispersie, cum ar fi MD5 or SHA
   b) C – identitatea clientului
   c) N_S – valoarea aleatoare generată de server
   d) N_C – valoarea aleatoare generată de client
   e) X – soluţia puzzle-ului

6. La primirea rezultatului, serverul:
   a) Calculează timpul necesar clientului pentru a rezolva puzzle-ul: $T_{solve} = T_s - T_R$, unde T_R este timpul de recepţie. Dacă acest timp este mai mic decât timpul estimat T_E, clientul are la dispoziţie o putere de calcul mare. În acest caz serverul poate să decidă terminarea comunicaţiei sau întârzierea deliberată a comunicaţiei.
   b) Verifică dacă clientul a mai furnizat o soluţie cu aceiaşi parametri N_S şi N_C, caz în care comunicaţia cu clientul C încetează.
   c) Verifică corectitudinea soluţiei **X**.
7. Dacă toate condiţiile sunt satisfăcute, serverul poate să aloce resurse pentru execuţia protocolului de autentificare a clientului curent.

3.1.7.4 Puzzle-uri adaptate clienţilor

Varietatea dispozitivelor care accesează diverse servicii este largă, pornind de la sisteme desktop de ultimă generaţie, laptop-uri în diverse configuraţii şi terminând cu dispozitive mobile cum ar fi PDA-uri şi smartphone-uri. Puterea de calcul variază în limite largi la aceste dispozitive şi în mod evident, în conjuncţie cu tehnologia threshold puzzles, experienţa utilizatorului ar fi foarte diferită în condiţii de încărcare similare. Acest lucru se întâmplă deoarece un sistem desktop sau laptop este mai puternic decât un PDA, iar în cazul autentificării la acelaşi server întârzierea observată de utilizator fiind mult mai mare în cazul PDA-ului.

Vorbim în acest caz de necesitatea adaptării puzzle-urilor la performanţele fiecărui client în parte, lucru ce ar duce în final la o experienţă uniformă a utilizatorului, indiferent de dispozitivul ales. Adaptarea dificultăţii puzzle-ului este o idee nouă căreia i-am dat un nume sugestiv: Adaptive Threshold Puzzles [12]. Ideea de bază din spatele acestui mecanism este determinarea puterii de calcul aproximative a clientului şi adaptarea proporţională a dificultăţii puzzle-ului în cazul unui atac.

La primul dialog între client şi server sau la cererea expresă a clientului poate avea loc determinarea puterii de calcul a clientului. Serverul va trimite un puzzle de explorare care conţine o problemă ce trebuie rezolvată de client, timpul de

rezolvare determinând puterea de calcul efectivă. Problema poate fi o inversare parţială a unei funcţii de dispersie (procedură asemănătoare celei din cazul client puzzles) sau poate fi o altă metodă total diferită, în funcţie de implementarea reală. Este recomandabil ca dificultatea şi timpul de rezolvare să aibă o dependenţă liniară.

Puterea de calcul P_C a clientului poate fi calculată de către server pe baza timpului în care clientul rezolvă problema. Se poate însă ca un client maliţios să întârzie în mod deliberat trimiterea răspunsului, pentru a ascunde adevărata valoare pentru P_C şi pentru ca serverul să-i trimită puzzle-uri cu o dificultate redusă în cazul unui atac, probabil lansat tot de către acesta. De aceea, trebuie găsită o metodă de a încuraja clientul să-şi folosească întreaga putere de calcul pe durata rezolvării puzzle-ului de explorare. O soluţie ar fi acordarea unui număr limitat de conexiuni pe unitatea de timp proporţional cu puterea de calcul estimată pentru acesta. Spre exemplu, unui client puternic i se alocă un număr maxim de N conexiuni pe unitatea de timp, iar unui client mai puţin puternic (cum ar fi un PDA sau un telefon mobil) i se vor acorda doar N/2 sau N/3 conexiuni pe aceeaşi unitate de timp. Dacă un client maliţios rezolvă puzzle-ul de explorare într-un timp mare, determinând astfel o putere P_C mai mică decât are de fapt, numărul de conexiuni pe unitatea de timp va fi proporţional mai mic, iar conexiunile care depăşesc acest număr sunt ignorate. În acest fel, atacurile unui client care nu-şi raportează corect puterea de calcul, nu sunt posibile.

Complexitatea adaptată *k* a puzzle-ului se poate calcula astfel:

$$k_C = round\left(k \cdot \log_2\left(\frac{P_C}{P_{ref}}\right)\right)$$

Ec. 2

Unde:

- P_{ref} – Puterea de calcul de referinţă
- k – Dificultatea de referinţă (corelată cu P_{ref})
- P_C – Puterea de calcul raportată de un client
- k_C – Dificultatea adaptată clientului

Dificultatea de referinţă se calculează ca şi în cazul threshold puzzles:

$k <= \log_2 (\max(1, Q * t_s / t_c)) + 1$, unde:

- Q – dimensiunea curentă a cozii de aşteptare
- t_s – timpul necesar efectuării unei operaţii criptografice, pentru server
- t_c – timpul necesar efectuării unei operaţii criptografice, pentru client

3.1.7.5 Algoritmul Adaptive Threshold Puzzles

Având în vedere motivarea din paragraful anterior, algoritmul Adaptive Threshold Puzzles este următorul:

1. La începerea comunicaţiei cu un client, serverul verifică starea sistemului. Dacă încărcarea nu depăşeşte un prag considerat critic, execuţia algoritmului se opreşte, deoarece sistemul de apărare nu este necesar.
2. Dacă clientul este întâlnit pentru prima oară, serverul trebuie să afle puterea de calcul aproximativă de care dispune clientul, trimiţând un puzzle de explorare. Acest puzzle este o simplă inversiune parţială a unei funcţii de dispersie. Puzzle-ul trebuie să aibă o complexitate medie, spre exemplu k = 6. Numărul de cereri permise de client per unitatea de timp vor fi în funcţie de puterea raportată de client în acest pas.
3. Serverul generează o valoare aleatoare şi unică $\mathbf{N_s}$ (nonce) cu o entropie de 64 de biţi şi memorează momentul de timp $\mathbf{T_S}$ la care generarea a avut loc.
4. Serverul stabileşte un nivel de dificultate $\mathbf{k_C}$ personalizat pentru puzzle şi estimează timpul minim $\mathbf{T_E}$ de rezolvare al acestuia, $k_C = round\left(k \cdot \log_2\left(\frac{P_C}{P_{ref}}\right)\right)$
5. Serverul creează un puzzle pentru client:

   « N_S, k, T_S »

   - N_S – valoarea aleatoare generată de server
   - k – nivelul de dificultate a puzzle-ului
   - T_S – timpul la care valoarea aleatoare a fost generată
6. La recepţia puzzle-ului, clientul verifică faptul că amprenta de timp T_S este recentă, generează un număr aleator N_C, şi rezolvă puzzle-ul aplicând în mod repetat o funcţie de dispersie unei cantităţi formate din C, N_S, N_C şi X. Puzzle-ul se consideră rezolvat când k biţi consecutivi ai cantităţii Y sunt egali cu 0.

   h(C, N_S, N_C, X) = Y

   f) h – funcţie criptografică de dispersie, cum ar fi MD5 or SHA
   g) C – identitatea clientului
   h) N_S – valoarea aleatoare generată de server
   i) N_C – valoarea aleatoare generată de client
   j) X – soluţia puzzle-ului
7. La primirea rezultatului, serverul:
   a) Calculează timpul necesar clientului pentru a rezolva puzzle-ul: $T_{solve} = T_s - T_R$, unde T_R este timpul de recepţie. Dacă acest timp este mai mic decât timpul estimat T_E, clientul are la dispoziţie o putere de calcul mare. În acest caz serverul poate să decidă terminarea comunicaţiei sau întârzierea deliberată a comunicaţiei.

b) Verifică dacă clientul a mai furnizat o soluţie cu aceiaşi parametri N_S şi N_C, caz în care comunicaţia cu clientul C încetează.
c) Verifică corectitudinea soluţiei **X**.

8. Se aplică constrângerea ca numărul de conexiuni de la client per unitatea de timp să nu depăşească un prag corespunzător puterii de calcul raportate de acesta la pasul 2.
9. Dacă toate condiţiile sunt satisfăcute, serverul poate să aloce resurse pentru execuţia protocolului de autentificare a clientului curent.

3.2 Aspecte practice de implementare

Aspectele teoretice expuse în paragrafele anterioare au fost validate în practică. Pentru a reduce timpul de dezvoltare al prototipului software şi pentru a elimina problemele inerente ale unor implementări de la zero, am utilizat biblioteca Mentalis Security Library [98] care este o implementare open-source a SSL şi TLS în .NET Framework.

3.2.1 Algoritmul SSL Handshake

Profitând de faptul că SSL este un protocol de autentificare popular, dată fiind prezenţa sa în orice browser, este oportun a verifica consideraţiile teoretice asupra acestuia. Pentru înţelegerea schimbărilor propuse în SSL este necesar să prezentăm întâi configuraţia sa nemodificată.

Protocolului SSL Handshake este descris în Fig. 15. Schimbul de mesaje între client şi server conţine informaţii referitoare la capabilităţile criptografice ale clientului precum şi alegerea serverului în această privinţă.

Se observă că mesajul CERTIFICATE conţine o semnătură electronică, serverul angajându-şi în acest fel resursele fără a cunoaşte identitatea clientului şi fără să ştie dacă acesta este bine intenţionat sau nu. Dacă clientul nu intenţionează să continue dialogul cu serverul ci dimpotrivă, doar să îl supraîncarce cu cereri inutile care vor fi semnate în cele din urmă, avem de-a face cu un atac DoS tipic.

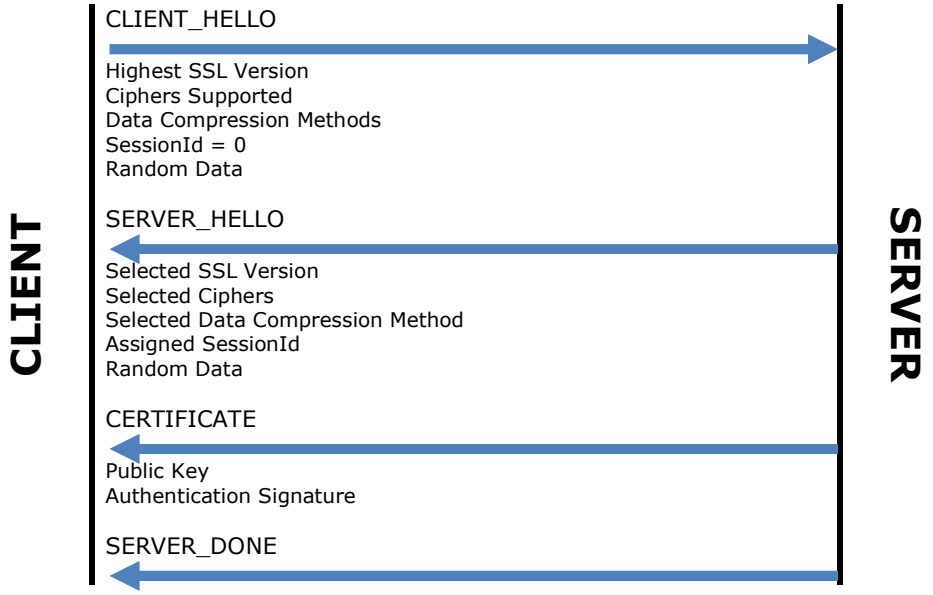


Fig. 15 Protocolul SSL Handshake

Din punct de vedere al disponibilităţii, angajarea necondiţionată de resurse din partea serverului este un neajuns. Tocmai de aceea am făcut propunerea de a utiliza tehnologiile Threshold Puzzles şi Adaptive Threshold Puzzles în cadrul sistemelor de autentificare în general şi mai cu seamă în cel mai răspândit sistem de autentificare existent, SSL. Modificările aduse protocolului nu afectează credibilitatea sa criptografică ci îi conferă posibilitatea de a egaliza efortul procesului de autentificare pe cele două părţi implicate, clientul şi serverul.

În cele ce urmează se va prezenta varianta modificată a protocolului, aşa cum a fost ea implementată în cadrul prototipului software.

3.2.2 Algoritmul SSL Handshake cu distribuţie egală de efort

Suportul pentru tehnologia Threshold Puzzles presupune modificarea fluxului de mesaje ale protocolului SSL Handshake şi anume introducerea a două mesaje adiţionale numite PUZZLE_CHALLENGE respectiv PUZZLE_SOLUTION înainte de mesajul SERVER_DONE care marchează sfârşitul protocolului. Scopul acestei modificări este distribuirea egală a efortului între cele două entităţi implicate în procesul de autentificare, clientul şi serverul. Cantitatea de procesare pe care trebuie să o efectueze serverul trebuie să se menţină sub un prag critic. Dacă pragul este depăşit este necesar ca avalanşa de cereri de autentificare să fie controlată, o metodă potrivită fiind ocuparea clientului cu o sarcină cu dificultate proporţională cu întârzierea necesară serverului pentru a ţine pasul.

Mesajul PUZZLE_CHALLENGE conţine timpul la care puzzle-ul a fost creat (*TimeStamp*), nivelul de dificultate al puzzle-ului (*Difficulty*) şi o valoare aleatoare unică aleasă de server (*ServerNonce*). Acest mesaj este trimis de server ca răspuns la mesajul CLIENT_HELLO numai în cazul în care încărcarea serverului depăşeşte un prag considerat critic. La rândul său clientul răspunde cu mesajul PUZZLE_SOLUTION, care conţine identitatea sa (*ClientID*), valoarea aleatoare de la server (*ServerNonce*), o valoare unică aleatoare generată de client (*ClientNonce*) şi soluţia puzzle-ului (*Solution*).

Fluxul mesajelor pentru algoritmul SSL Handshake cu distribuţie egală de efort este prezentat în Fig. 16:

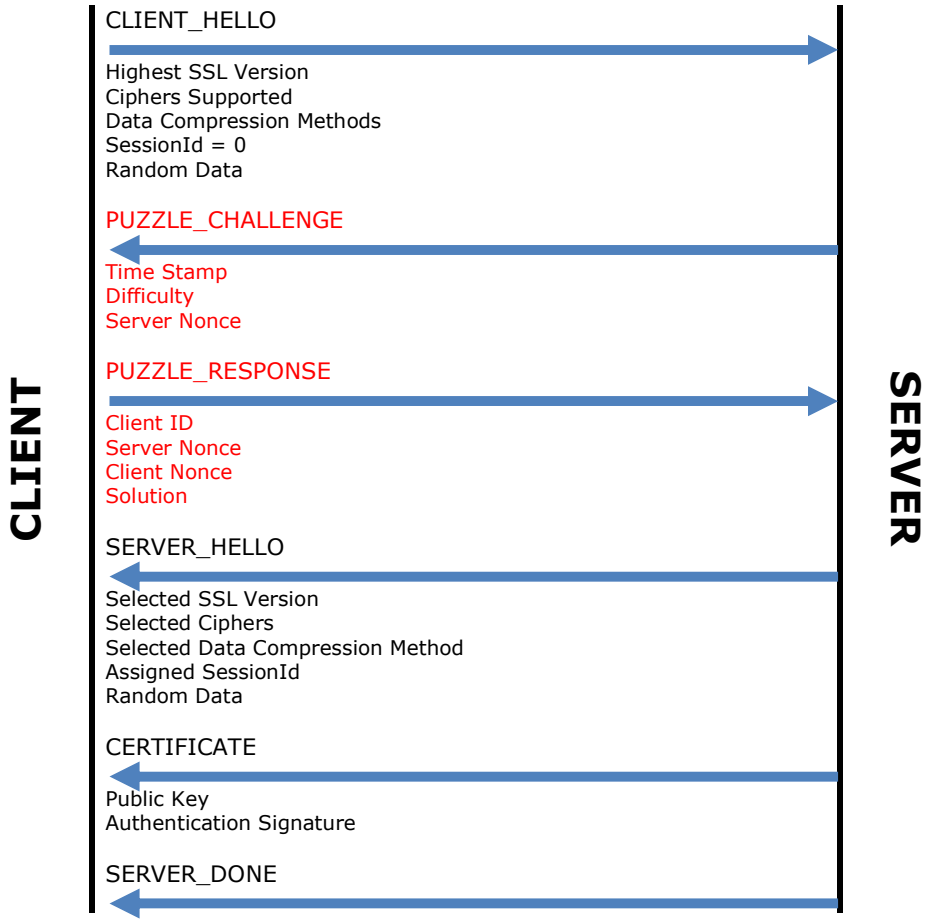


Fig. 16 Protocolul SSL Handshake cu distribuţie egală de efort

Adăugarea celor două mesaje suplimentare (PUZZLE_CHALLENGE şi PUZZE_SOLUTION) nu schimbă integritatea criptografică a protocolului SSL ci are ca scop ameliorarea disponibilităţii serverului prin întârzierea clientului cu o durată de timp proporţională cu încărcarea curentă a serverului. Schimbările la nivel criptografic ar presupune o abordare diferită şi deosebit de laborioasă pentru a evita introducerea de noi vulnerabilităţi.

3.2.3 Algoritmul SSL Handshake cu distribuţie adaptivă de efort

Pentru scenariile în care clienţii unui sistem sunt eterogeni din punct de vedere al capacităţii de calcul (laptop-uri, PDA-uri, etc.) se impune adoptarea tehnologiei Adaptive Threshold Puzzles. La fel ca în cazul distribuţiei egale de efort din paragraful anterior este necesară modificarea fluxului de mesaje ale protocolului SSL Handshake prin introducerea de mesaje adiţionale. Două dintre mesaje sunt cele pe care le-am amintit deja, şi anume PUZZLE_CHALLENGE respectiv PUZZLE_SOLUTION şi au aceleaşi funcţii. În plus, distribuţia adaptivă a efortului necesită cunoaşterea puterii de calcul a clientului, acest lucru fiind aflat prin altă pereche de mesaje: PUZZLE_EXPLORE_CHALLENGE şi PUZZLE_EXPLORE_SOLUTION. Scopul acestei modificări este distribuirea efortului între cele două entităţi implicate în procesul de autentificare, clientul şi serverul, însă proporţional cu puterea de calcul a clientului. Cantitatea de procesare pe care trebuie să o efectueze serverul trebuie să se menţină sub un prag critic. Dacă pragul este depăşit este necesar ca avalanşa de cereri de autentificare să fie controlată, o metodă potrivită fiind ocuparea clientului cu o sarcină cu dificultate proporţională cu întârzierea necesară serverului pentru a ţine pasul.

Dacă clientul care solicită autentificarea este necunoscut, serverul are nevoie să cunoască puterea sa de calcul. Această determinare se face prin solicitarea rezolvării unui puzzle de explorare cu dificultate medie, prin mesajul PUZZLE_EXPLORE_CHALLENGE. Răspunsul la acest mesaj vine din partea clientului prin mesajul PUZZLE_EXPLORE_SOLUTION. Dacă clientul este cunoscut (a mai efectuat cereri de autentificare în trecut) se poate omite pasul de explorare pentru un timp nedeterminat sau pentru un timp stabilit în mod unilateral de către server. Odată cunoscută puterea de calcul a clientului (fie din explorarea curentă sau una petrecută anterior), serverul poate cere clientului rezolvarea unui puzzle, asemenea cu cazul anterior.

Mesajele PUZZLE_EXPLORE_CHALLENGE şi PUZZLE_CHALLENGE conţin timpul la care puzzle-ul a fost creat (*TimeStamp*), nivelul de dificultate al puzzle-ului (*Difficulty*) şi o valoare aleatoare unică aleasă de server (*ServerNonce*). Clientul răspunde cu mesajele PUZZLE_EXPLORE_SOLUTION şi PUZZLE_SOLUTION, care conţin identitatea sa (*ClientID*), valoarea aleatoare de la server (*ServerNonce*), o valoare unică aleatoare generată de client (*ClientNonce*) şi soluţia puzzle-ului (*Solution*).

Fluxul mesajelor pentru algoritmul SSL Handshake cu distribuţie adaptivă de efort este prezentat în Fig. 17:

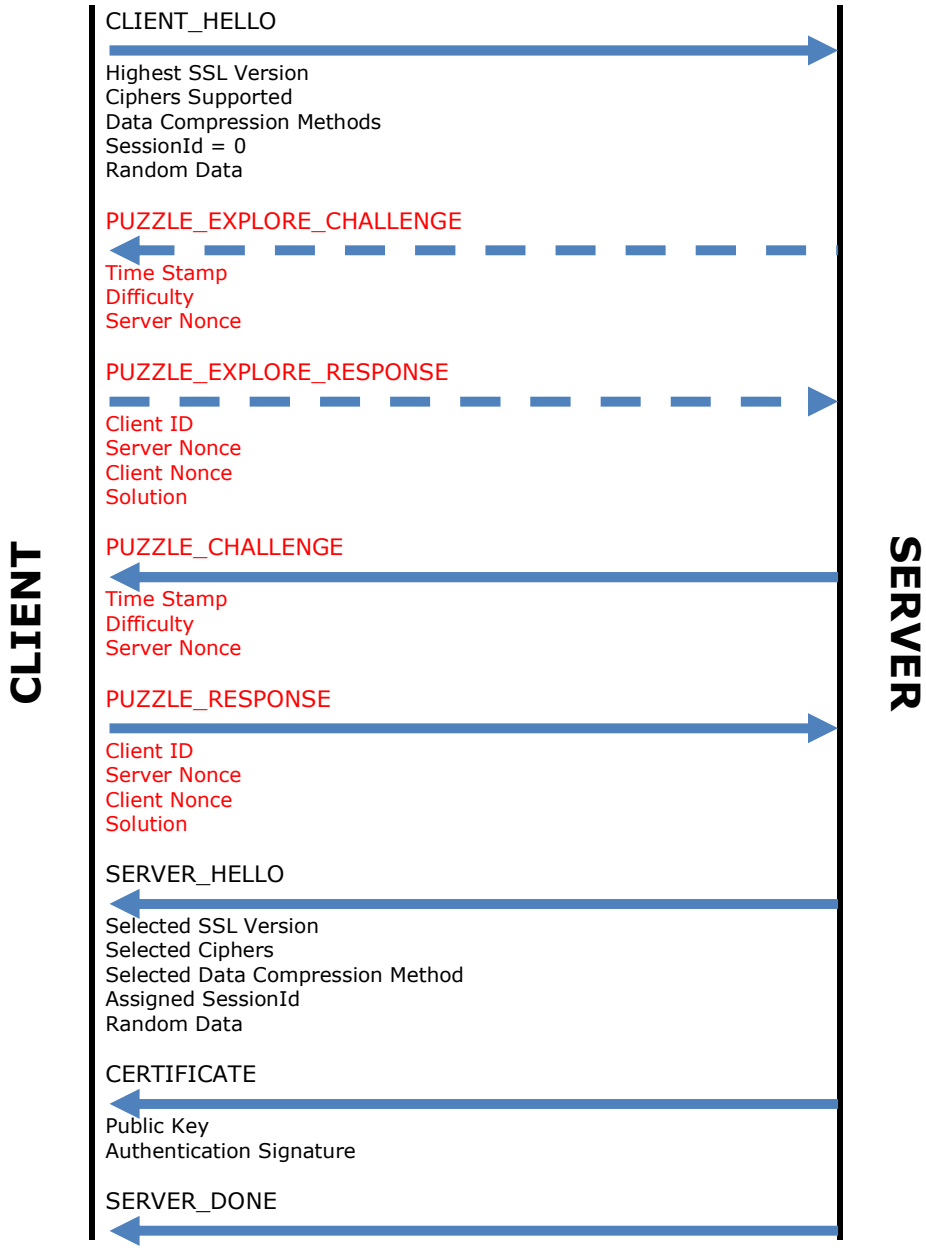


Fig. 17 Protocolul SSL Handshake cu distribuţie adaptivă de efort

La fel ca în cazul anterior, adăugarea mesajelor suplimentare nu schimbă integritatea criptografică a protocolului SSL ci are ca scop ameliorarea disponibilităţii serverului prin întârzierea clientului cu o durată de timp proporţională cu încărcarea curentă a serverului. Schimbările la nivel criptografic ar presupune o abordare diferită şi deosebit de laborioasă pentru a evita introducerea de noi vulnerabilităţi.

Spunem că distribuţia efortului este adaptivă deoarece întârzierea clientului se face în funcţie de puterea de calcul pe care o are.

3.2.4 Prototipul software

Pentru a simula comportarea tehnologiei Threshold Puzzles într-un context real şi pentru a evidenţia avantajele acesteia faţă de tehnologia Client Puzzle propusă în literatură, s-a realizat un prototip cu structura indicată în Fig. 18.

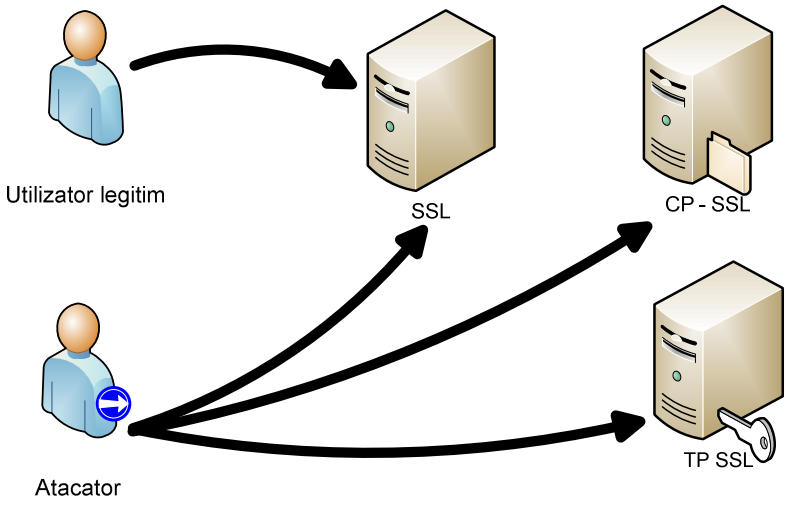


Fig. 18 Schema prototipului Threshold Puzzles

Prototipul conţine următoarele module:

- **Utilizator legitim** – un client obişnuit, care doreşte efectuarea unei autentificări SSL şi urmează execuţia protocolului conform cu specificaţia.

- **Atacator** – un client cu acces la o putere de calcul mare şi care efectuează un număr mare de cereri false de autentificare într-un timp scurt.
- **Server SSL** – Un server SSL obişnuit şi neprotejat.
- **Server CP SSL** – Un server SSL protejat cu tehnologia *Client Puzzles*.
- **Server TP SSL** – Un server SSL protejat cu tehnologia *Threshold Puzzles*.

Întrucât în condiţii de laborator accesul la o putere de calcul semnificativă este dificil, pentru a simula acest lucru s-a convenit ca modulul reprezentând atacatorul să emită o soluţie aleatoare – şi evident incorectă – urmând ca verificarea pe partea serverului să nu se efectueze. În acest fel atacatorul pare că rezolvă puzzle-ul într-un timp considerabil mai mic decât clienţii legitimi. Această modificare nu este în măsură să afecteze în mod semnificativ rezultatele experimentului.

Simularea s-a efectuat pe două sisteme de calcul legate într-o reţea de 100 Mb/s prin care nu a existat trafic perturbator în timpul desfăşurării simulării. Pe sistemul reprezentând serverul au fost instalate modulele *Server SSL*, *Server CP SSL* şi *Server TP SSL*. Pe sistemul reprezentând clientul au fost instalate modulele *Utilizator legitim* şi *Atacator* apoi s-au lansat instanţe multiple ale acestora pentru a simula existenţa mai multor clienţi.

3.2.5 Performanţă

Pe sistemul de test au fost rulate mai multe scenarii menite să determine comportarea tehnologiilor Client Puzzles şi Threshold Puzzles. Aceste scenarii sunt descrise în cele ce urmează:

3.2.5.1 Clienţi autentificaţi de server normal

Pentru a avea un etalon în măsurarea performanţei, s-a rulat sistemul în scenariul clasic, adică clienţi obişnuiţi autentificaţi de un server SSL normal. În acest fel s-a putut determina timpul mediu necesar pentru execuţia completă a protocolului SSL, fără adăugirile specifice Client/Threshold Puzzles.

S-a efectuat un număr de 100 de autentificări a 15 clienţi, timpul mediu de răspuns al serverului fiind de 4545 ms.

3.2.5.2 Clienţi şi atacator autentificaţi de server normal

În această simulare s-au folosit 14 clienţi normali şi un atacator. Scopul atacatorului este de a trimite o rafală de mesaje CLIENT_HELLO către server în scopul de a declanşa un număr egal de răspunsuri semnate de server. Execuţia

protocolului nu este finalizată niciodată de către atacator, iar răspunsul serverului este ignorat.

Atacatorul a trimis cereri false de autentificare cu o rată de 30 pe secundă, timp în care încărcarea pe server a fost de 100%. Timpul mediu de autentificare a clienţilor legitimi a fost de 11320 ms.

3.2.5.3 Clienţi şi atacator autentificaţi de server cu Client Puzzles

Această simulare este asemănătoarea cu cea anterioară, cu excepţia serverului căruia i s-a activat tehnologia Client Puzzles. Protocolului SSL i s-au adăugat practic două mesaje (numite PUZZLE_CHALLENGE respectiv PUZZLE_RESPONSE) aşa cum s-a descris în paragrafele anterioare, în speranţa reducerii numărului de cereri venite pe unitatea de timp de la un client.

Deoarece clientul are acces la o putere de calcul semnificativă, fapt simulat prin generarea unui rezultat aleatoriu (evident, incorect) şi lipsa verificării corectitudinii sale pe partea de server, serverul a fost inundat cu cereri de autentificare false, cu aceeaşi rată de 30 pe secundă ca în cazul anterior. În tot acest timp încărcarea pe server a fost de 100% iar timpul mediu de autentificare a clienţilor legitimi a fost de 12730 ms. Timpul uşor mai mare decât în cazul precedent indică un overhead mai mare din cauza celor două mesaje suplimentare din protocol.

3.2.5.4 Clienţi şi atacator autentificaţi de server cu Threshold Puzzles

Simularea anterioară a fost reluată modificând configuraţia serverului SSL să folosească tehnologia Threshold Puzzles. În acest caz, timpul mediu de autentificare al clienţilor legitimi a fost de 4553 ms, aproximativ egal cu cazul în care serverul nu se află sub atac. Acest rezultat se datorează faptului că atacatorul a rezolvat mult prea repede şi în mod repetat puzzle-urile solicitate de server, ceea ce a dus automat la blocarea comunicaţiei cu clientul atacator.

3.2.5.5 Concluzie

În timpul unui aşa-numit atac puternic, tehnologia Client Puzzles nu a avut efectul scontat deoarece atacatorul este capabil a rezolva puzzle-urile intr-un timp foarte scurt şi poate încă trimite un număr considerabil de cereri false serverului. Timpul mediu de execuţie al protocolului a fost similar cu cel din cazul unui server neprotejat. Un server protejat cu Threshold Puzzles însă a refuzat sistematic cererile însoţite de soluţii sosite prea rapid pentru nivelul de dificultate ales, astfel încât timpul de autentificare perceput de clienţi a rămas practic acelaşi cu cel din condiţii normale.

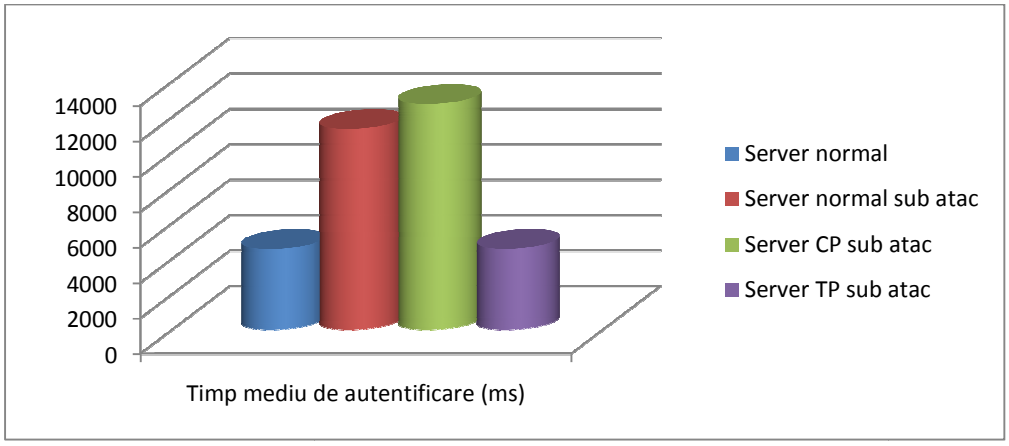


Fig. 19 Comparaţie a rezultatelor simulărilor pe protocolul SSL

3.3 Detectarea şi ameliorarea locală a atacurilor DoS

Pentru ca măsurile luate de sistemele de autentificare în vederea ameliorării rezistenţei la atacuri DoS să fie eficiente, este necesar ca acestea să cunoască dacă încărcarea sistemului este datorată unui factor temporar şi tranzitoriu, unui vârf de sarcină sau într-adevăr, există un atac în desfăşurare. Desfăşurarea de măsuri de apărare în timpul unor situaţii tranzitorii introduce de cele mai multe ori o penalitate percepută de clienţi ca o degradare a performanţei, iar situaţia este cu atât mai neplăcută cu cât acestea apar mai des.

Datorită naturii complexe a sistemelor de autentificare şi datorită multitudinii de configuraţii în care acestea pot activa, detectarea atacurilor este o problemă relativ dificilă în lipsa unor metrici standard de evaluare a riscului. În cadrul paragrafului de faţă ne propunem analiza unui sistem de autentificare de tip Single Sign-On, şi propunerea unei metrici şi unui algoritm de estimare a riscului. Am ales pentru exemplificare suita de protocoale Liberty [55][56].

Sistemele de autentificare de tip Single Sign-On (SSO) în general şi implementarea în suita Liberty în particular sunt vulnerabile la atacuri DoS. Cerinţa ca fiecare mesaj între entităţile sistemului să fie semnat permite unui atacator să inunde reţeaua cu mesaje false pe care entităţile se vor grăbi să le decodifice, doar ca să constate ulterior că mesajul provine de la un client necunoscut, semnătura este invalidă şi resursele consumate pentru verificarea semnăturii s-au pierdut inutil. În Fig. 20 se arată mecanismul de principiu al unui atac DoS îndreptat asupra sistemelor SSO. În cazul unui atac, clienţii legitimi ai furnizorului de identitate aflat

sub atac vor constata degradarea timpului de răspuns la solicitările lor datorită încărcării artificiale pe furnizorul de identitate [13].

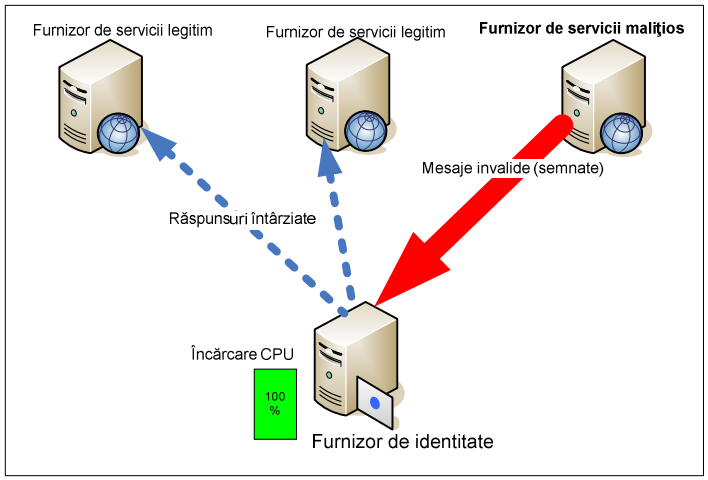


Fig. 20 Atac DoS asupra sistemelor SSO (în particular implementarea Liberty)

Simpla semnare a fiecărui mesaj vehiculat reprezintă în sine un atac DoS. Deşi uneori sistemele SSO sunt guvernate şi de înţelegeri legale, acest lucru nu este suficient pentru a proteja utilizatorii legitimi de atacuri, fiind nevoie de adoptarea unor măsuri de natură tehnică. Dacă în paragrafele anterioare s-a discutat despre oportunitatea adoptării unor măsuri care să încarce clienţii proporţional cu serverul (vezi tehnologiile Threshold Puzzles şi Adaptive Threshold puzzles), acum vom vorbi despre metode prin care sistemul de autentificare îşi dă seama de un atac în desfăşurare, fiind capabil să se adapteze situaţiilor existente în diverse scenarii folosind inferenţa bayesiană [23], tehnică folosită şi la filtrele spam din sistemele de poştă electronică.

Inferenţa bayesiană este o alegere potrivită pentru evaluarea stării unui sistem de autentificare sau a unui sistem SSO. Starea sistemului se evaluează şi se estimează pe baza unui set de senzori software [79] plasaţi în anumite puncte din sistem, iar informaţiile culese de aceştia sunt agregate la un nivel central care determină gradul de probabilitate ca sistemul supervizat să fie sub atac. În funcţie de acest grad de probabilitate se poate lua măsura de a varia plaja de dificultate pentru puzzle.

3.3.1 Caracteristici măsurabile

Serverul care asigură serviciul de autentificare trebuie să aibă o măsură exactă a încărcării momentane sau din viitorul apropiat. Întrucât în literatura de specialitate nu am găsit caracteristici măsurabile ale protocoalelor Liberty, am

enumerat cele mai importante caracteristici cu potenţial asupra sistemului hardware. Caracteristicile se folosesc ca date de intrare pentru motorul de inferenţă bayesiană care la rândul său va determina nivelul de siguranţă al sistemului SSO.

Caracteristicile măsurabile ale protocoalelor Liberty pot fi diferenţiate în mai multe categorii descrise în continuare, în funcţie de domeniul vizat.

3.3.1.1 Disciplina de execuţie a protocolului

O primă caracteristică măsurabilă din această categorie se referă la ordinea de execuţie a protocoalelor din suita Liberty. Execuţia trebuie să urmeze o ordine aproximativă, un deranj al acestei ordini putând fi semnul unui posibil atac asupra sistemului SSO. Spre exemplu, o cerere *Single Logout* nu ar trebui primită de un furnizor de servicii sau de identitate dacă cererea *Single Sign-On and Federation* a fost primită anterior, cum se vede în Fig. 21 [13].

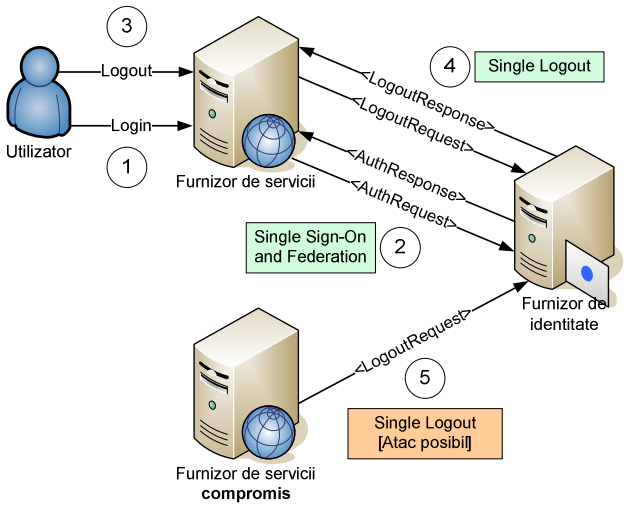


Fig. 21 Detectarea unui posibil atac asupra sistemelor SSO folosind ordinea de execuţie a protocolului.

Conţinutul cererilor este o altă caracteristică de luat în considerare. Pentru fiecare protocol, cererea are un set de atribute care specifică cum ar trebui procesată aceasta de către entitatea receptoare. O variaţie largă a acestor atribute din partea unui client anume indică faptul că acesta este neglijent în formularea mesajelor introducând probabil date aleatoare, fapt ce indică un atac împotriva serverului. Spre exemplu, mesajul *AuthRequest* din cadrul protocolului *Single Sign-On and Federation* are un atribut care cere furnizorului de identitate să emită un identificator anonim şi temporar în numele clientului, când se schimbă datele între

furnizorii de servicii. Prea multe mesaje *AuthRequest* cu acest atribut setat, recepţionate de la acelaşi client, ar putea indica un posibil atac DoS asupra furnizorului de identitate.

3.3.1.2 Sănătatea reţelei

Caracteristicile legate de reţea fac parte din această categorie şi constau în principal din parametrii traficului pe partea de recepţie [5]:

- **Anomalii în operarea reţelei** – aceste tipuri de anomalii includ diferenţele semnificative în comportarea reţelei sau funcţionare la limita fizică a hardware-ului. Anomaliile se pot distinge printr-o creştere bruscă, aproape verticală a valorilor măsurate pe o perioadă de timp.
- **Anomalii Flash Crowd** – acest tip de anomalie se datorează comportamentului sinergic al unui grup de utilizatori, spre exemplu ca urmare a lansării unui produs software nou. De obicei anomaliile de acest fel se manifestă în puncte bine stabilite din reţea cum ar fi locaţiile de download sau mirror-urile.
- **Anomalii de abuz** – sunt datorate fie unor atacuri DoS în desfăşurare (prin inundarea reţelei) sau datorită unei activităţi de scanare a porturilor. Anomalia se deosebeşte de operarea normală a reţelei şi de anomalia Flash Crowd, dar nu întotdeauna este evidentă. Totuşi, prin detectarea şi măsurarea unor modele de trafic această anomalie poate fi descoperită.

Un senzor software plasat în sistem trebuie să monitorizeze anomaliile din reţea şi să furnizeze aceste informaţii în formă brută sau semi-agregată motorului de detecţie a atacurilor. Anumiţi parametri (cum ar fi timpul dintre două cereri consecutive, mărimea cererii, numărul de cereri pe secundă de la un client) au fost deja folosiţi cu succes în detectarea atacurilor DoS [79][49], însă nu au fost agregaţi într-un sistem complex de evaluare a stării de sănătate a unui server de autentificare.

3.3.1.3 Sănătatea sistemului gazdă

Nivelul curent de sănătate al sistemului gazdă este un indiciu crucial asupra calităţii serviciului furnizat. În condiţii normale de funcţionare şi încărcare, sistemul poate oferi servicii cu un factor QoS stabilit prin design. Alterarea condiţiilor de natură software sau hardware are ca efect modificarea QoS cu consecinţe asupra clienţilor care constată modificări în calitatea serviciului furnizat de sistem, cum ar fi: întârzieri în furnizarea răspunsului, răspuns cu intermitenţă, lipsa totală a răspunsului, etc.

Sănătatea sistemului gazdă depinde de factori cum ar fi:

- Cantitatea de memorie disponibilă
- Încărcarea curentă a procesorului, în scenarii uniprocesor
- Încărcarea curentă a procesorului pentru care serviciul are afinitate, în scenarii multiprocesor
- Activitatea I/O a discurilor
- Rezerva de resurse alocate maşinii virtuale, pentru scenariile virtuale
- Starea şi nivelul de activitate al driverelor de sistem, servicii, daemons, etc.

3.4 Detecţia euristică a atacurilor cu SSO-SENSE

Informaţiile colectate de senzorii plasaţi în diverse puncte ale sistemului monitorizat sunt disponibile la un moment dat unui modul software specializat. Acesta poate agrega informaţiile în aşa fel încât să tragă o concluzie legată de starea momentană a sistemului, în scopul luării măsurilor de apărare împotriva atacurilor sau de limitare ale consecinţelor acestora.

Având ca bază teoria inferenţei bayesiene, am creat un modul de estimare a nivelului de risc, numit SSO-SENSE. Schema sa de principiu este indicată în Fig. 22.

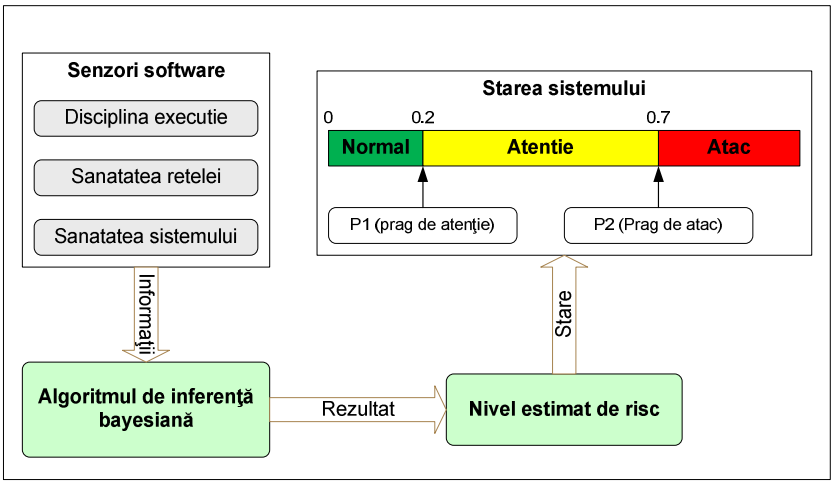


Fig. 22 Modul de estimare a nivelului de risc SSO-SENSE

Senzorii software sunt responsabili cu colectarea informaţiilor din sistemul supravegheat. Acei senzori care se ocupă de disciplina de execuţie a protocolului trebuie implementaţi direct în cadrul protocoalelor SSO, în aşa fel încât să aibă în mod nemijlocit acces la schimbul de mesaje dintre părţi. Senzorii responsabili de sănătatea reţelei şi a sistemului pot utiliza indicatorii de performanţă existenţi în

cadrul sistemului de operare, în cazul Windows putându-se utiliza *Windows Management Instrumentation* (WMI).

Informaţiile colectate de senzori sunt agregate de algoritmul de inferenţă bayesiană, din agregare rezultând un nivel estimat de risc. Acest nivel este un număr real între 0 şi 1. Valoarea 0 indică funcţionarea normală a sistemului cu un grad ridicat de certitudine. O valoare care se apropie de 1 indică prezenţa unui număr de anomalii în cadrul sistemului, ale căror ponderi pot indica un atac în desfăşurare.

Nivelurile pentru pragul de atenţie respectiv cel de atac se aleg astfel încât $0 < P1 < P2 < 1$, cu P1 şi P2 spaţiate în cadrul intervalului astfel încât să asigure o delimitare eficientă a acţiunilor care se petrec în starea de atenţie respectiv starea de atac.

3.4.1 Scurtă fundamentare matematică

Pentru a înţelege modalitatea de agregare a informaţiilor primite de la senzori, în acest paragraf se face o scurtă fundamentare matematică a teoriei inferenţei bayesiene.

Considerăm un set de trei ipoteze reciproc exclusive H_1, H_2 şi H_3 corespunzătoare stărilor în care se poate afla sistemul – Normal, Atenţie, Atac. De asemenea, considerăm un eveniment E care apare când un senzor software detectează o condiţie de prag între caracteristicile măsurabile ale protocolului de autentificare.

Bazându-ne pe considerentele anterioare, se poate calcula *constanta de normalizare* Λ astfel:

$$\Lambda = P(E \mid H_1) \cdot P(H_1) + P(E \mid H_2) \cdot P(H_2) + P(E \mid H_3) \cdot P(H_3)$$

Ec. (3)

Iniţial se consideră cu un grad mare de siguranţă că sistemul nu este sub atac, putem deci asocia câte o probabilitate pentru fiecare dintre stări:

$P(H_1) = 0.9$ (sistem în stare normală)

$P(H_2) = 0.09$ (sistem în stare de atenţie)

$P(H_3) = 0.01$ (sistem sub atac).

La apariţia unui eveniment E, evaluăm probabilităţile posterioare are celor trei ipoteze, astfel:

$$P(H_1 \mid E) = \frac{P(E \mid H_1) \cdot P(H_1)}{\Lambda}$$

Ec. (4)

$$P(H_2 \mid E) = \frac{P(E \mid H_2) \cdot P(H_2)}{\Lambda}$$

Ec. (5)

$$P(H_3 \mid E) = \frac{P(E \mid H_3) \cdot P(H_3)}{\Lambda}$$

Ec. (6)

Cu fiecare eveniment E care se produce în sistem, probabilităţile posterioare ale ipotezelor H_1, H_2 şi H_3 se modifică. Starea momentană a sistemului după producerea evenimentului E este:

$$S = MAX [P(H_1 \mid E), P(H_2 \mid E), P(H_3 \mid E)]$$

Ec. (7)

Evenimentul E poate fi:

- Încărcarea disciplinei de execuţie a protocolului de autentificare sau SSO, inversiunea de mesaje, mesaje lipsă sau cu parametri incompleţi.
- Timpul între două cereri succesive.
- Eroare de autentificare.
- Creştere bruscă a încărcării pe CPU pentru o perioadă scurtă de timp.
- Creştere bruscă a traficului pe reţea, exceptând anomalii flash crowd.
- Încercarea de acces la porturi successive (scanare).

3.4.2 Aplicarea detecţiei euristice în autentificare

Diversele metode de apărare împotriva atacurilor DoS aplicate sistemelor de autentificare îşi pot îmbunătăţi comportamentul şi deciziile prin informaţiile suplimentare oferite de sistemele de detecţie ale atacurilor. În timp ce un sistem complex de detecţie a intruşilor aplicat la nivelul global al reţelei poate ajuta într-o manieră generală, detecţia atacurilor la nivelul vectorului de autentificare este o problemă specifică. În acest caz atacatorul nu urmăreşte pătrunderea în sistem şi accesul la date confidenţiale ci compromiterea în sine a autentificării şi împiedicarea accesului clienţilor legitimi, fapt ce nu interesează sistemele de *intrusion detection* existente actual.

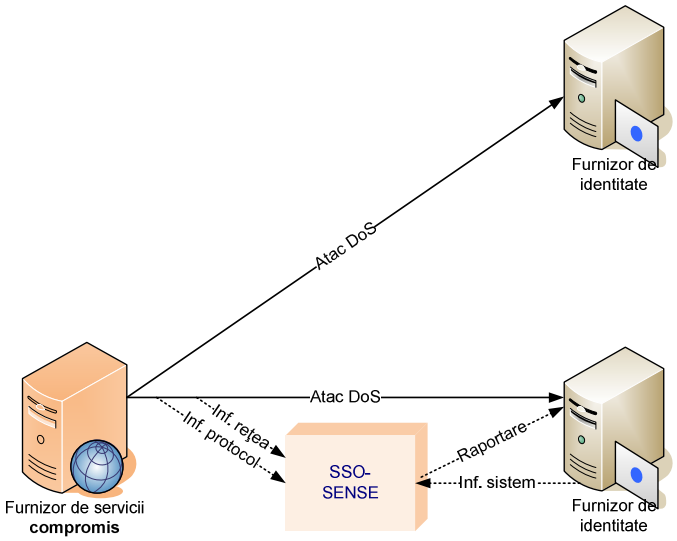


Fig. 23 Folosirea detecţiei euristice la nivelul vectorului de autentificare

În Fig. 23 un furnizor de servicii compromis din cadrul unui sistem SSO lansează un atac asupra a doi furnizori de identitate. Ambii furnizori sunt protejaţi împotriva atacurilor de tip DoS, cu diferenţa că unul beneficiază de un sistem de detecţie euristică ce îi permite să ia măsuri mai eficiente şi posibil cu impact inferior la nivelul clienţilor legitimi.

Conform Ec. 1, nivelul de dificultate k al unui threshold puzzle se calculează astfel:

$$k_{max} = \log_2 (\max(1, Q * t_s / t_c)) + 1$$

În lipsa informaţiilor despre iminenţa unui atac, dificultatea puzzle-ului poate fi aleasă proporţional cu lungimea cozii cererilor sau poate urma o regulă de creştere dinainte stabilită. Dacă informaţia cu privire la starea probabilă a sistemului este disponibilă , regula de variaţie a parametrului k poate fi:

- **Stare normală** – serverul nu ajunge la încărcarea maximă decât în puţine moment de vârf. Este nevoie preponderant de puzze-uri cu complexitate redusă, care se poate alege în mod neuniform din plaja $[0; k_{max}]$.
- **Stare de atenţie** – încărcarea pe server depăşeşte constant un prag de confort, fără ca acest lucru să indice automat un atac în desfăşurare. Este nevoie preponderent de puzzle-uri de complexitate medie din plaja $[0; k_{max})$.

- **Stare de atac** – există indicii că un atac este în desfăşurare. Pe durata acestei stări serverul poate lua deciza ca nivelul de dificultate k_{max} să fie menţinut, chiar dacă serverul nu a ajuns la încărcarea maximă.

3.4.3 Rezultate experimentale

Prototipul modulului de estimare a nivelului de risc SSO-SENSE a fost implementat în limbajul C#, pe platforma .NET Framework 3.5. Pentru a determina comportamentul modulului de inferenţă bayesiană, scenariul de test a constat în simularea a două situaţii care pot fi întâlnite în mod real:

- Autentificări eşuate repetate – sistemul este bombardat cu cereri de autentificare false, despre care atacatorul ştie că vor eşua, cu scopul de a încărca serverul.
- Neurmarea cursului firesc al protocolului – aceasta urmăreşte destabilizarea serverului prin răspunsuri nepotrivite cu specificaţia, lucru care lasă serverul în aşteptare până la un eventual timeout.

La simulare s-a convenit ca starea iniţială a sistemului să fie cea normală, încărcarea curentă pe server Q = 400000, cu t_s = 0,003 şi t_c = 0,5. Dificultatea maximă a puzzle-ului este astfel k_{max} = 12.

Autentificările eşuate sunt evenimente cu gravitate redusă, de aceea tranziţia de la starea normală la starea de atac se face trecând prin mai multe stări de atenţie. Nivelul de dificultate determinat de mecanismul puzzle k_{max} este coroborat cu starea curentă a sistemului. Simularea unui şir de autentificări eşuate duce la următoarea ieşire a prototipului:

```
Status before Authentication Failure: Normal
Status after failure no. 1: Normal (k = 1)
Status after failure no. 2: Warning (k = 4)
Status after failure no. 3: Warning (k = 5)
Status after failure no. 4: Warning (k = 4)
Status after failure no. 5: Warning (k = 4)
Status after failure no. 6: Attack (k = 12)
```

Se observă că după primul eveniment de autentificare eşuată, sistemul rămâne în starea normală, deoarece acest tip de evenimente pot fi provocate în mod curent de utilizatori, fără că acest lucru să însemne neapărat un semn de atac. Următoarele evenimente însă provoacă trecerea sistemului în starea de atenţie şi în cele din urmă în starea de atac, semnalizând administratorului gravitatea situaţiei.

Evenimentele de tip eroare de protocol sunt mai grave (lucru stabilit iniţial prin probabilităţile asociate ipotezelor) şi deci implicit convergenţa sistemului către starea de atac este mai rapidă. Rezultatul rulării în condiţiile a 6 evenimente succesive de acest tip este dat mai jos:

```
Status before Protocol Followup Failure: Normal
Status after failure no. 1: Warning (k = 6)
Status after failure no. 2: Warning (k = 4)
Status after failure no. 3: Warning (k = 5)
Status after failure no. 4: Attack (k = 12)
Status after failure no. 5: Attack (k = 12)
Status after failure no. 6: Attack (k = 12)
```

3.5 Concluzii

Una dintre soluţiile propuse în literatură pentru ameliorarea rezistenţei la atacuri DoS a protocoalelor de autentificare a fost folosirea tehnologiei client puzzles cu scopul de a creşte artificial costul de execuţie al protocolului pe client, proporţional cu creşterea încărcării pe server în scopul realizării unei bucle de autoreglare. Cu toate acestea însă, datorită naturii paralelizabile mecanismul client puzzles poate fi învins în anumite condiţii de performanţă ale atacatorului.

Pentru a evita situaţiile în care un atacator reuşeşte să rezolve puzzle-urile într-un timp suficient de scurt pentru a fi capabil de a trimite o rafală de cereri către server, în esenţă crescând nivelul de dificultate al puzzle-ului şi blocând astfel clienţii legitimi cu calcule inutile, am adus două modificări de design, grupate sub denumirea „Threshold Puzzles”, după cum urmează:

- Limitarea superioară a nivelului de dificultate al puzzle-ului – pentru ca un eventual atac puternic să nu aibă ca rezultat indirect un atac DoS asupra clienţilor înşişi.
- Stabilirea unui timp minim de răspuns – estimarea timpului minim necesar unui client mediu de a rezolva puzzle-ul.

Pentru scenariile în care clienţii unui serviciu sunt eterogeni, având puteri de calcul ce variază în limite largi, tehnologia a fost şi mai mult rafinată prin adaptarea încărcării artificiale a clientului în funcţie de puterea sa reală de calcul. Acest lucru a pornit de la observaţia că un PDA sau Smartphone au puteri de calcul mult mai mici decât un laptop sau un desktop. În condiţii de încărcare mare a serverului, când valoarea dificultăţii puzzle-ului este mare, clienţii cu putere de calcul redusă vor întâmpina dificultăţi în rezolvare, ceea ce este o degradare a experienţei utilizatorului. Astfel, am creat conceptul de „Adaptive Threshold Puzzles” care – aşa cum îi spune şi numele oferă spre rezolvare clienţilor puzzle-uri cu dificultăţi personalizate fiecărui client în parte.

În urma experimentelor realizate cu prototipul implementat am demonstrat că un server protejat cu Threshold Puzzles se comportă în termeni comparabili cu un server în lipsa atacului.

Măsurile de apărare împotriva atacurilor DoS pe care un sistem le are la dispoziţie sunt eficace atunci când acesta cunoaşte faptul că un atac este în desfăşurare. Pentru a determina acest lucru nu este suficientă o metrică simplă, care să implice doar câţiva parametri sistem cum ar fi încărcarea memoriei şi a procesorului, deoarece în acest fel am avea de-a face cu numeroase alarme false. Este nevoie de un modul care să analizeze informaţiile disponibile la un moment determinat şi să ia decizii în consecinţă, evitând catalogarea evenimentelor tranzitorii drept atacuri. Pentru aceasta am propus folosirea inferenţei bayesiene ca bază pentru un modul de detecţie a nivelului de risc numit SSO-SENSE.

Contribuţiile aduse de acest capitol pot fi sumarizate după cum urmează:

- Au fost dezvoltate tehnologiile Threshold Puzzles şi Adaptive Threshold Puzzles, pentru a acoperi situaţiile în care tehnologia Client Puzzles nu este eficientă.
- S-a arătat practic rezistenţa la atacuri DoS a protocolului SSL, căruia i s-a adaptat tehnologia Threshold Puzzles.
- S-a evidenţiat posibilitatea ca atacurile DoS să fie detectate euristic prin inferenţă bayesiană, ca metodă care vizează vectorul de autentificare al unui sistem.
- Nivelul de dificultate al unui puzzle poate fi ales optim în funcţie de starea sistemului de autentificare determinată euristic, adică niveluri de dificultate preponderent reduse pentru un sistem neatacat şi niveluri preponderent ridicate sau nivelul maxim pentru un sistem aflat sub atac.

4. Contribuţii la creşterea disponibilităţii reţelelor tip GSM

Suntem cvasidependenţi de comunicaţie. Fie că e vorba de mediul personal, cel de afaceri, medical sau de altă natură, comunicaţia joacă un rol însemnat în viaţa de zi cu zi a societăţii. Comunicaţiile critice – adică cele care au loc în urgenţele medicale, în situaţii de criză sau calamităţi naturale – au loc în majoritatea cazurilor pe o infrastructură care nu a fost concepută pentru disponibilitate crescută sau redundanţă. Un eventual atac asupra infrastructurii de comunicaţie – mobilă sau fixă – poate avea un impact nedorit. Deoarece cel mai răspândit standard de comunicaţie mobilă este GSM, în acest capitol se va aborda problema disponibilităţii sale, problemele pe care le are design-ul actual precum şi modalităţile prin care se pot ameliora efectele atacurilor DoS. Deşi tehnologia GSM are o vârstă şi tinde să fie înlocuită treptat de 3G, WiMax sau LTE, subiectul este de actualitate întrucât noile tehnologii vor asigura acoperire doar local, în timp ce suprafeţele mari de teritoriu sunt acoperite de GSM.

Problemele de securitate ale GSM sunt numeroase, însă o încercare de a le rezolva pe toate în acelaşi timp ar fi sortită eşecului. Se impune astfel o clasificare a vulnerabilităţilor după anumite criterii din care să rezulte o prioritizare în acţiune. În capitolul de faţă se face o astfel de analiză pentru ca mai apoi să se prezinte soluţii la cea mai gravă problemă cu care se confruntă tehnologia GSM.

4.1 Atacuri asupra reţelelor GSM

Atacurile asupra unui sistem de comunicaţii cum este GSM sunt numeroase, complexe şi nu întotdeauna evidente. Multitudinea de factori care influenţează comunicaţia, complexitatea modelelor implicate în tehnologie, factorul uman, factorul social, sunt lucruri care afectează într-un fel sau altul siguranţa comunicaţiei. Literatura menţionează un număr de atacuri posibile asupra sistemelor GSM [38], însă nu indică gravitatea şi probabilitatea acestor atacuri. Simpla apreciere subiectivă a gravităţii nu este o cale precisă, de aceea este necesar un oarecare grad de formalism pentru determinarea celor mai grave şi posibile atacuri.

În [16] s-a propus aplicarea unei metodologii de clasificare a atacurilor GSM, metodologie derivată din software şi care poartă numele DREAD. Aceasta a fost introdusă de Howard şi LeBlanc în [46].

4.1.1 Clasificarea DREAD a vulnerabilităţilor

Nivelul de risc al unei vulnerabilităţi, fie la nivel software fie la nivel de infrastructură de comunicaţii, se poate determina formal având în vedere anumiţi factori, după cum urmează:

- **Potenţialul de distrugere –** Cât de mare este pierderea dacă vulnerabilitatea este exploatată? Se măsoară distrugerea maximă posibilă şi se acordă un scor de la 1 la 10. Scorul 10 este o ameninţare care permite atacatorului să ocolească toate mecanismele de protecţie şi să acţioneze în voie în sistemul compromis.
- **Reproductibilitatea –** Cât de uşor se poate reproduce atacul? Se măsoară uşurinţa cu care o ameninţare devine exploatabilă, pe o scară de la 1 la 10. O reproductibilitate înaltă este în interesul oricărui atacator.
- **Exploatabilitatea –** Cât de uşor este ca atacul să fie lansat? Spre exemplu, dacă un programator începător cu un PC acasă poate iniţia atacul, scorul acestui factor ar fi clar 10. Dacă însă atacul ar necesita o agenţie guvernamentală care ar trebui să investească o sută de milioane de dolari, scorul acestui factor ar trebui să fie 1. De asemenea, se va estima ce nivel de autentificare şi autorizare sunt necesare pentru a ataca sistemul. Spre exemplu, dacă un utilizator anonim de la distanţă poate ataca sistemul, scorul ar fi 10, spre deosebire de un utilizator local căruia i se cer credenţiale puternice, caz în care scorul ar fi mult mai mic.
- **Utilizatori afectaţi –** Câţi utilizatori sunt afectaţi? Aceasta înseamnă aproximativ procentajul de utilizatori care ar fi impactaţi de un atac: 91-100% (scor 10), pana la 0-10% (scor 1). Trebuie de asemenea gândit şi în termeni de număr absolut de utilizatori. Un singur procent din 100 de milioane de utilizatori înseamnă totuşi o mulţime de utilizatori.
- **Uşurinţa în descoperire –** Cât de uşor se poate descoperi vulnerabilitatea? Aceasta este probabil cea mai dificil de determinat metrică şi se va presupune întotdeauna ca având valoarea 10, deoarece mai devreme sau mai târziu o vulnerabilitate va fi descoperită şi făcută publică.

4.1.2 Clasificarea vulnerabilităţilor reţelelor GSM după sistemul DREAD

În literatura de specialitate nu există o clasificare a vulnerabilităţilor GSM după gravitate. În lucrări precum [38] se pune accent doar pe aspectele tehnice ale atacurilor, fără a ţine seama de experienţa tehnică a atacatorului şi resursele necesare pentru a-l pune în practică. Lucrarea de faţă însă abordează într-o manieră comparativă vulnerabilităţile din GSM şi le clasifică în funcţie de probabilitatea de exploatare. Această abordare permite prioritizarea măsurilor de ameliorare a securităţii.

Evaluarea scorului de risc s-a făcut după cum urmează: pentru fiecare vulnerabilitate în parte s-a acordat câte un punctaj de la 1 la 10 fiecărui factor din

sistemul DREAD (potenţial de distrugere, reproductibilitate, exploatabilitate, utilizatori afectaţi, uşurinţa în descoperire). Media aritmetică a celor 5 punctaje dă scorul de risc.

4.1.2.1 Denial of Service

Interfaţa radio GSM este vulnerabilă la atacuri DoS deoarece resurse preţioase cum sunt canalele de semnalizare sunt oferite oricui le solicită. Inundarea canalelor de semnalizare cu cereri legitime sau false înseamnă în esenţă că traficul este paralizat. Inundarea canalului de semnalizare poate fi cauzată de un atacator cu o staţie mobilă modificată [14] sau de cereri legitime [34].

Factor	Scor de risc
Potenţial de distrugere	5
Reproductibilitate	10
Exploatabilitate	7
Utilizatori afectaţi	9
Uşurinţă în descoperire	10
Factor de risc "Denial of Service"	8.2

4.1.2.2 De-registration Spoofing

Un atacator poate falsifica o cerere de de-înregistrare către reţea (IMSI detach). Aceasta înseamnă că utilizatorul este detaşat din aria vizitată şi astfel toate serviciile terminate la mobil vor fi afectate.

Factor	Scor de risc
Potenţial de distrugere	3
Reproductibilitate	10
Exploatabilitate	5
Utilizatori afectaţi	1
Uşurinţă în descoperire	10
Factor de risc "De-registration Spoofing"	5.8

4.1.2.3 Location Update Spoofing

Acest atac este similar cu cel anterior. Un atacator trimite o cerere falsă de actualizare a locaţiei către o zonă diferită de cea în care utilizatorul se află. Ca şi în cazul precedent, serviciile terminate la mobil vor fi afectate.

Factor	Scor de risc
Potenţial de distrugere	3
Reproductibilitate	10
Exploatabilitate	5
Utilizatori afectaţi	1
Uşurinţă în descoperire	10
Factor de risc "Location Update Spoofing"	5.8

4.1.2.4 Camping on a False BTS

Terminalul mobil poate fi „momit" să se campeze pe un BTS fals, făcându-l astfel inaccesibil semnalelor de apel ale reţelei mobile. Alternativ, BTS-ul fals poate acţiona ca un intermediar şi ar putea permite traficul după bunul plac. Acest atac necesită un BTS modificat.

Factor	Scor de risc
Potenţial de distrugere	3
Reproductibilitate	10
Exploatabilitate	4
Utilizatori afectaţi	1
Uşurinţă în descoperire	10
Factor de risc "Camping on a False BTS"	5.6

4.1.2.5 Passive Identity Caching

În anumite condiţii, reţeaua poate cere utilizatorului să își decline identitatea într-o manieră clară, necriptată. Un terminal mobil modificat poate fi folosit pentru a memora informaţia pentru utilizări ulterioare.

Factor	Scor de risc
Potenţial de distrugere	2
Reproductibilitate	8
Exploatabilitate	5
Utilizatori afectaţi	1
Uşurinţă în descoperire	10
Factor de risc "Passive Identity Caching"	5.2

4.1.2.6 Active Identity Caching

Acest atac este similar cu cel anterior, cu excepţia faptului că utilizatorul este „momit" să se campeze pe un BTS fals care la rândul său va cere în permanenţă ca identitatea să fie vehiculată în clar.

Factor	Scor de risc
Potenţial de distrugere	2
Reproductibilitate	8
Exploatabilitate	4
Utilizatori afectaţi	1
Uşurinţă în descoperire	10
Factor de risc "Active Identity Caching"	5

4.1.2.7 Encryption Suppression

Deoarece staţia mobilă nu are cum să autentifice mesajele de pe interfaţa radio, acesta poate fi „momit" să se campeze pe un BTS fals şi să comunice cu atacatorul într-o manieră necriptată. Atacatorul poate elimina comanda de criptare şi menţine convorbirea necriptată pentru atât timp cât aceasta rămâne nedetectată.

Factor	Scor de risc
Potenţial de distrugere	2
Reproductibilitate	10
Exploatabilitate	3

Utilizatori afectaţi	1
Uşurinţă în descoperire	10
Factor de risc "Encryption Suppression"	5.2

4.1.2.8 Compromised Cipher Key

Acest atac necesită un BTS modificat şi posesia de către intrus a unui vector de autentificare compromis, exploatând astfel vulnerabilitatea că utilizatorul nu are control asupra cheii de cifrare. Utilizatorul ţintă este „momit" să se campeze pe un BTS/MS fals. Când se efectuează un apel, perechea BTS/MS falsă forţează folosirea cheii de cifrare compromisă.

Factor	Scor de risc
Potenţial de distrugere	2
Reproductibilitate	8
Exploatabilitate	3
Utilizatori afectaţi	1
Uşurinţă în descoperire	10
Factor de risc "Compromised Cipher Key"	4.8

4.1.2.9 Eavesdropping on User Data by Suppressing Encryption

Acest atac necesită un BTS/MS modificat care exploatează vulnerabilitatea că staţia mobilă nu poate autentifica mesajele prin reţeaua radio. Utilizatorul este tentat să se campeze pe un BTS fals, iar când acesta încearcă să iniţieze un apel, criptarea nu este activată prin falsificarea mesajelor care cer acest lucru. Atacatorul poate însă să creeze o legătură reală şi securizată cu reţeaua, permiţând trecerea traficului. În acest fel atacatorul poate să captureze date de trafic necodificate.

Factor	Scor de risc
Potenţial de distrugere	2
Reproductibilitate	10
Exploatabilitate	2
Utilizatori afectaţi	1
Uşurinţă în descoperire	10

Factor de risc "Eavesdropping on User Data by Suppressing Encryption"	5

$Risk_{DREAD}$ = (2 + 10 + 2 + 1 + 10) / 5 = 5

4.1.2.10 Suppression of Encryption between Target User and True Network

Utilizatorul este tentat să se campeze pe un BTS/MS fals. Când utilizatorul sau reţeaua legitimă iniţiază o conexiune, BTS/MS-ul fals modifică parametrii de criptare ai MS-ului pentru a face să pară că există o incompatibilitate între reţea şi staţia mobilă. Reţeaua poate decide să stabilească o conexiune necifrată. După ce decizia a fost luată, intrusul poate să captureze traficul în voie.

Factor	Scor de risc
Potenţial de distrugere	2
Reproductibilitate	10
Exploatabilitate	2
Utilizatori afectaţi	1
Uşurinţă în descoperire	10
Factor de risc "Suppression of Encryption between target User and True Network"	5

4.1.2.11 Eavesdropping on User Data by Forcing the Use of a Compromised Cipher Key

Acest atac necesită un BTS/MS modificat şi posesia de către intrus a unui vector de autentificare compromis, exploatând astfel slăbiciunea că utilizatorul nu are un control asupra cheii de cifrare. Utilizatorul ţintă este tentat să se campeze pe BTS/MS-ul fals. Când utilizatorul ţintă sau intrusul iniţiază un serviciu, BTS/MS-ul fals forţează utilizarea unei chei de cifrare compromise pe partea mobilă, în timp ce creează o conexiune cu reţeaua reală folosind un abonament real.

Factor	Scor de risc
Potenţial de distrugere	2
Reproductibilitate	10
Exploatabilitate	2
Utilizatori afectaţi	1
Uşurinţă în descoperire	10

Factor de risc "Eavesdropping on User Data by Forcing the Use of a Compromised Cipher Key"	5

4.1.2.12 User impersonation with compromised authentication vector

Acest atac necesită un MS modificat şi posesia de către intrus a unui vector de autentificare compromis care va fi utilizat de către reţea pentru a autentifica utilizatorul legitim. Intrusul foloseşte aceste informaţii pentru a impersona utilizatorul ţintă în reţeaua legitimă.

Factor	Scor de risc
Potenţial de distrugere	2
Reproductibilitate	10
Exploatabilitate	2
Utilizatori afectaţi	1
Uşurinţă în descoperire	10
Factor de risc "User impersonation with compromised authentication vector"	5

4.1.2.13 User impersonation through eavesdropped authentication response

Atacul necesită un MS modificat şi exploatează slăbiciunea că un vector de autentificare se poate folosi de mai multe ori. Intrusul poate captura răspunsul de autentificare trimis de utilizator şi foloseşte acest răspuns când aceeaşi întrebare este trimisă ulterior. Intrusul foloseşte răspunsul capturat pentru a impersona utilizatorul ţintă în reţea.

Factor	Scor de risc
Potenţial de distrugere	2
Reproductibilitate	10
Exploatabilitate	5
Utilizatori afectaţi	1
Uşurinţă în descoperire	10
Factor de risc "User impersonation through eavesdropped authentication response"	5.6

4.1.2.14 Hijacking outgoing calls in networks with encryption disabled

Acest atac necesită un BTS/MS modificat. În timp ce utilizatorul ţintă se campează pe un BTS fals, intrusul semnalizează utilizatorul ţintă că are un apel de intrare. Acesta la rândul să iniţiază procedura de stabilire a apelului, pe care intrusul o permite să se efectueze între utilizator şi reţea, modificând elementele de semnalizare în aşa fel încât pentru reţea să pară că utilizatorul doreşte să iniţieze un apel. Reţeaua nu activează criptarea. După autentificare, intrusul taie conexiunea cu utilizatorul şi în continuare foloseşte conexiunea cu reţeaua pentru a efectua apeluri frauduloase pe abonamentul utilizatorului.

Factor	Scor de risc
Potenţial de distrugere	4
Reproductibilitate	10
Exploatabilitate	5
Utilizatori afectaţi	1
Uşurinţă în descoperire	10
Factor de risc "Hijacking outgoing calls in networks with encryption disabled"	6

4.1.2.15 Hijacking outgoing calls in networks with encryption enabled

Acest atac necesită un BTS/MS modificat. În plus faţă de atacul anterior, de această dată intrusul trebuie să încerce să suprime criptarea prin modificarea mesajelor prin care MS-ul informează reţeaua despre capacităţile sale criptografice.

Factor	Scor de risc
Potenţial de distrugere	4
Reproductibilitate	10
Exploatabilitate	5
Utilizatori afectaţi	1
Uşurinţă în descoperire	10
Factor de risc "Hijacking outgoing calls in networks with encryption enabled"	6

4.1.2.16 Hijacking incoming calls in networks with encryption disabled

Acest atac necesită un BTS/MS modificat. În timp ce utilizatorul ţintă se campează pe o staţie de bază falsă, un asociat al intrusului efectuează un apel la

numărul utilizatorului ţintă. Intrusul acţionează ca un intermediar între reţea şi utilizatorul ţintă până la momentul ulterior autentificării şi stabilirii apelului dintre aceştia. Reţeaua nu activează criptarea. După autentificare şi stabilirea apelului, intrusul eliberează utilizatorul ţintă şi foloseşte conexiunea pentru a răspunde la apelul efectuat de asociatul său. Utilizatorul ţintă va trebui să plătească pentru partea de apel din roaming.

Factor	Scor de risc
Potenţial de distrugere	4
Reproductibilitate	10
Exploatabilitate	5
Utilizatori afectaţi	1
Uşurinţă în descoperire	10
Factor de risc "Hijacking incoming calls in networks with encryption disabled"	6

4.1.2.17 Hijacking incoming calls in networks with encryption enabled

Acest atac necesită un BTS/MS modificat. În plus faţă de atacul anterior, intrusul va suprima criptarea.

Factor	Scor de risc
Potenţial de distrugere	4
Reproductibilitate	10
Exploatabilitate	5
Utilizatori afectaţi	1
Uşurinţă în descoperire	10
Factor de risc "Hijacking incoming calls in networks with encryption enabled"	6

4.1.3 Clasificarea vulnerabilităţilor după gravitate

Aplicarea metodologiei DREAD de clasificare a vulnerabilităţilor, metodă folosită iniţial în software, a permis ordonarea acestora în funcţie de gravitate (vezi Tabel **2**). Se constată astfel că cele mai grave vulnerabilităţi ale reţelelor GSM se referă la atacurile DoS şi posibilitatea ca apelurile să fie „furate" la momentul iniţierii lor. Atacurile DoS sunt relativ uşor de pus în practică deoarece tehnologia GSM nu are rezistenţă prin design împotriva atacurilor de acest tip.

Tabel 2. Clasificarea vulnerabilităţilor în GSM

Vulnerabilitate	Scor de risc
Denial of Service Attacks	8.2
Hijacking outgoing calls in networks with encryption disabled	6
Hijacking outgoing calls in networks with encryption enabled	6
Hijacking incoming calls in networks with encryption disabled	6
Hijacking incoming calls in networks with encryption enabled	6
De-registration Spoofing	5.8
Location Update Spoofing	5.8
Camping on a False BTS	5.6
User impersonation through eavesdropped authentication response	5.6
Passive Identity Caching	5.2
Encryption Suppression	5.2
Active Identity Caching	5
Eavesdropping on User Data by Suppressing Encryption	5
Suppression of Encryption between Target User and True Network	5
Eavesdropping on User Data by Forcing the Use of a Compromised Cipher Key	5
User impersonation with compromised authentication vector	5
Compromised Cipher Key	4.8

4.2 Atacuri DoS în reţele GSM

Am văzut că atacurile DoS sunt cea mai gravă ameninţare la adresa disponibilităţii sistemului GSM. Partea cea mai vulnerabilă a sistemului şi locul unde atacurile se desfăşoară este domeniul radio, mai precis sistemul de management al resurselor radio. Atacul de tip DoS în GSM are loc prin iniţierea de către un mobil a unui apel fals care are ca scop alocarea unor resurse radio. Repetarea acestui tip de apel duce la epuizarea resurselor şi la imposibilitatea folosirii serviciului de către un utilizator legitim. În acest paragraf se descrie în detaliu mecanismul care face GSM-ul vulnerabil în faţa unui atac DoS.

Scenariul tipic pentru partea preliminară a unui apel cu originea la mobil este după cum urmează [45]:

Pasul 1: Staţia mobilă (MS) cere acordarea unui canal de control de la BSC (Base Station Controller) (Fig. 24).

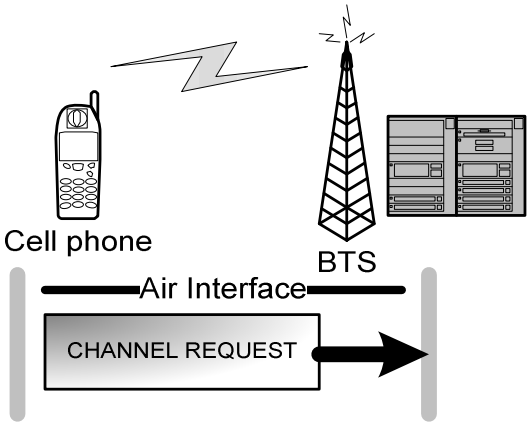


Fig. 24 Mesajul CHANNEL REQUEST

Pasul 2: BTS-ul decodifică mesajul CHANNEL REQUEST, calculează avansul temporal (distanţa MS□BTS) şi trimite informaţia completă la BSC prin intermediul mesajului CHANNEL REQUIRED. Totodată, se indică şi tipul de serviciu solicitat. (Fig. 25)

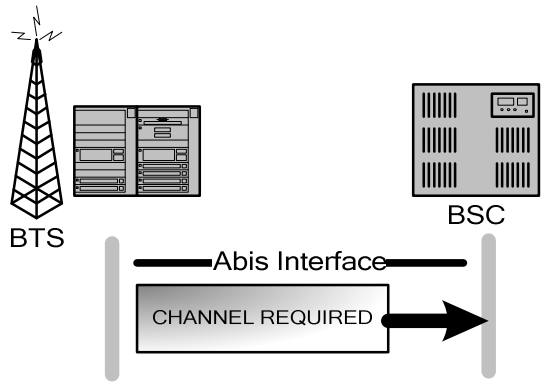


Fig. 25 Mesajul CHANNEL REQUIRED

Pasul 3: După recepţia şi procesarea mesajului CHANNEL REQUIRED, BSC-ul informează BTS-ul de ce tip de canal şi ce număr de canal va fi rezervat de un mesaj CHANNEL ACTIVE ulterior (Fig. 26).

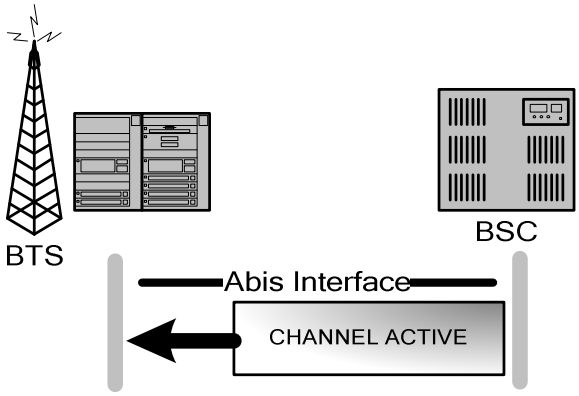


Fig. 26 Mesajul CHANNEL ACTIVE

Pasul 4: BTS-ul confirmă primirea prin trimiterea unui mesaj CHANNEL ACTIVE ACKNOWLEDGE (Fig. 27).

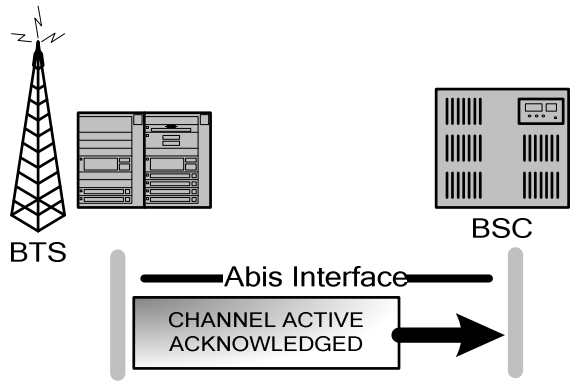


Fig. 27 Mesajul CHANNEL ACTIVE ACKNOWLEDGED

Pasul 5: BSC-ul trimite mesajul IMMEDIATE ASSIGNMENT COMMAND către BTS, care la rândul său informează MS-ul despre canalul alocat (Fig. 28).

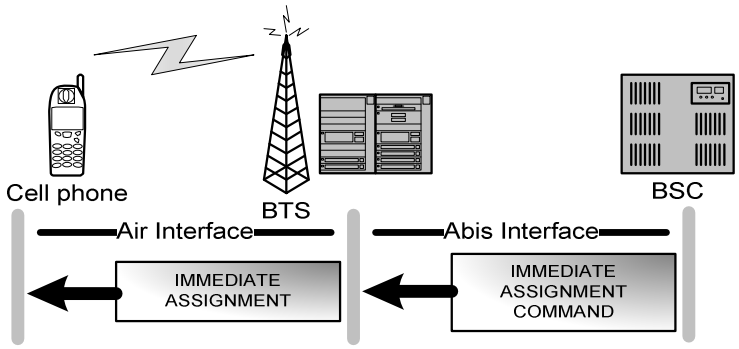


Fig. 28 Message IMMEDIATE ASSIGNMENT

Schimbul complet de mesaje al procesului de alocare de canal este arătat în Fig. 29.

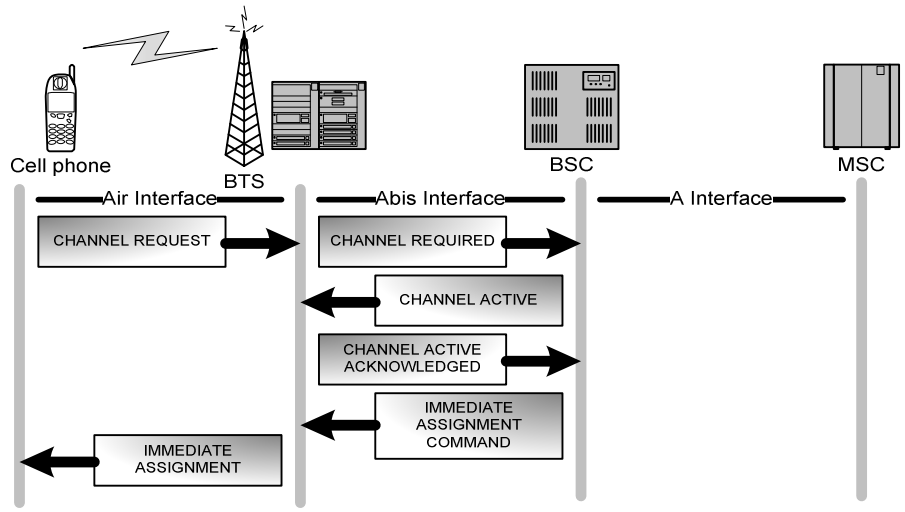


Fig. 29 Procesul de alocare a canalului în GSM

La sfârşitul procesului de alocare a canalului, ca urmare a cererii unei staţii mobile **neautentificate** (client din perspectiva noastră), BSC-ul a alocat un canal de semnalizare din grupul de canale disponibile. Staţia mobilă este acum responsabilă pentru respectarea restului protocolului, primul pas fiind cererea unui tip de serviciu.

Design-ul se bazează pe faptul că staţia mobilă va urma în mod corect fiecare pas din protocol. Ce se întâmplă însă dacă o staţie mobilă repetă scenariul anterior şi cere mai multe canale de semnalizare fără a continua protocolul până la capăt? Din moment ce numărul de canale de semnalizare este limitat, reţeaua

devine congestionată local şi cererile legitime sunt refuzate datorită lipsei de canale disponibile. BSC-ul va termina în cele din urmă cererile incomplete, eliberând resursele, dar acest lucru nu este un management de resurse eficient (resource containment) deoarece atacul în sine nu este detectat. Canalele de trafic disponibile nu vor deservite clienţilor legitimi deoarece toate canalele de semnalizare vor fi indisponibile (Fig. 30).

Chiar daca reţeaua ar efectua o minimă autentificare a staţiei mobile prin cererea numărului IMEI şi a puterii celor şase celule învecinate, atacul tot ar fi posibil întrucât staţia mobilă are control complet asupra acestor elemente şi ar putea raporta valori eronate pentru nivelurile de putere, generând în acelaşi timp numere IMEI sau redându-le dintr-o listă precompilată. Pentru a preîntâmpina problemele reţeaua are nevoie de o metodă simplă de verificare a identităţii clienţilor.

De remarcat este că îmbunătăţirea securităţii SIM-ului nu este o măsură judicioasă. În timp ce aplicarea unui sistem de securitate bazat pe SIM ar induce costuri reduse de rezolvare a vulnerabilităţii, soluţia ar fi inefectivă întrucât atacul este îndreptat asupra protocolului de stabilire a apelului şi nu asupra SIM-ului.

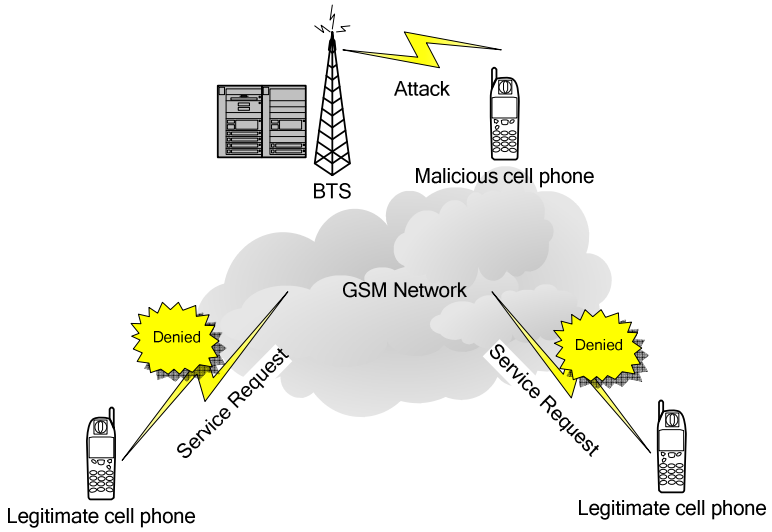


Fig. 30 Atacuri Denial of Service (DoS) în reţele GSM

4.3 Profilul atacatorului

Istoria recentă a înregistrat multe încercări de a proteja informaţia prin ascunderea ei sau cel puţin prin îngreunarea căilor de a o descoperi. Acest procedeu

este cunoscut sub numele de „securitate prin obscuritate” şi este cea mai mare sursă de probleme pentru orice sistem care încearcă să-şi securizeze datele. Mecanismul de securitate va fi în cele din urmă descoperit, comunicat grupurilor de interese şi exploatat.

Cei mai mulţi dintre indivizii care atacă sisteme de calculatoare sau de telefonie au motivaţii personale. Unii dintre ei vor încerca să exploateze slăbiciunile pentru câştiguri personale (acces gratuit la resurse, apeluri la distanţă gratuite, apeluri gratuite la numere cu valoare adăugată, etc.) în timp ce alţii vor prezenta slăbiciunile grupurilor interesate pentru faimă. Deşi atacurile sunt serioase şi pot provoca pierderi operatorului de reţea, atacurile Denial of Service sunt de departe cea mai gravă ameninţare. Un individ care atacă o organizaţie vulnerabilă poate paraliza traficul în arii extinse ale reţelei, cauzând pierderi financiare greu de estimat.

Pentru a ataca cu succes o reţea GSM, intrusul trebuie să fie capabil de a modifica codul firmware din telefon într-o manieră potrivită. Această sarcină nu este una simplă întrucât necesită cunoştinţe temeinice ale implementării particulare a dispozitivului mobil, codului şi chestiunilor tehnice specifice GSM. Acesta este aspectul cel mai provocator şi din fericire nu multe persoane au o asemenea experienţă.

Intrusul trebuie de asemenea să înţeleagă topologia reţelei pe care o atacă. Dacă aria ţintită este locală, o simplă plimbare prin oraş este suficientă pentru a determina locurile aproximative unde BTS-urile sunt localizate şi unde staţiile mobile maliţioase vor opera. Intrusul nu trebuie să fie în mod necesar prezent, întrucât telefonul mobil ar putea fi preprogramat să lanseze atacul, peisajul oraşului oferind o multitudine de locuri unde s-ar putea ascunde astfel de dispozitive mici.

Intrusul trebuie să fie motivat, în funcţie de magnitudinea atacului. Este improbabil ca un singur individ poate să fie motivat îndeajuns pentru a încerca un atac semnificativ asupra unei reţele GSM, în special considerând costul mare implicat. Totuşi, intrusul poate fi sprijinit de grupuri criminale, caz în care economia atacurilor are o altă magnitudine.

4.4 Economia atacului

Datorită cvasi-universalităţii şi cotei foarte mari de piaţă a reţelelor GSM, atacurile care ar putea să le afecteze sunt foarte tentante pentru organizaţiile criminale. Economia atacurilor asupra reţelelor GSM nu este diferită de cea a atacurilor îndreptate împotriva reţelelor de calculatoare, după cum urmează:

- Atacurile care cauzează oprirea totală a serviciului produc pierderi uriaşe de venit, fără a menţiona latura socială a acestora.

- Când comunicaţia este vitală, spre exemplu după un atac terorist sau dezastru natural, atacul DoS împotriva reţelelor GSM poate avea consecinţe dintre cele mai grave. Lipsa comunicaţiei în astfel de momente poate duce la pierderi de vieţi omeneşti sau proprietate.
- Atacurile care cauzează pauze în furnizarea serviciilor GSM sunt foarte dificil de detectat. Un client care doreşte să utilizeze reţeaua poate fi frustrat de imposibilitatea efectuării apelurilor. Dacă situaţia se repetă, încrederea în operatorul de telefonie poate fi afectată iar clienţii pot trece uşor la concurenţă.

4.5 Creşterea disponibilităţii reţelelor GSM prin preautentificare

Dacă cineva ar veni la uşa dumneavoastră şi v-ar cere un bun de valoare, i-aţi da acel lucru fără a-i stabili mai întâi identitatea? Aţi asuma implicit că oricine bate la uşă este un individ onest? Oricât de greu vă vine să credeţi, exact acest lucru se întâmplă la iniţierea unui apel într-o reţea GSM. În procesul de iniţiere a unui apel, în timpul VEA (*Very Early Assignment*) nu se efectuează autentificare sau cifrare [2].

Primul mesaj trimis de telefonul mobil este CHANNEL REQUEST şi are lungimea de doar 1 octet. Acesta conţine motivul cererii (răspuns la paging, apel de urgenţă, etc.) şi un identificator al tipului de canal pe care staţia mobilă îl preferă. Problema cu acest tip de abordare este că BSC-ul îşi alocă resursele valoroase unei staţii mobile neautentificate care ar putea să acţioneze în afara standardului.

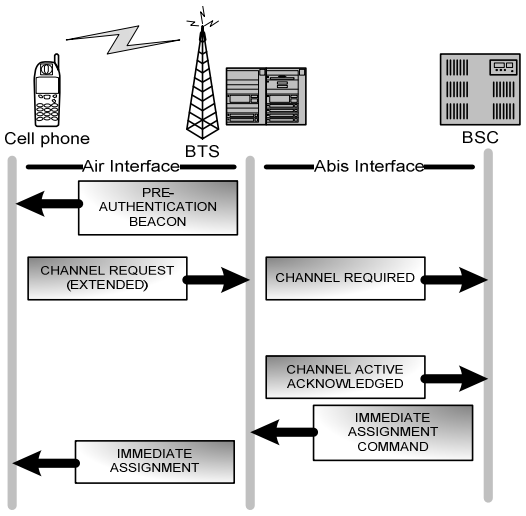


Fig. 31 Proces de asignare a canalului în GSM rezistent la atacuri DoS

Pentru a preveni un atac DoS potenţial, trebuie să existe o formă minimă de autentificare la momentul cererii canalului de comunicaţie, prin urmare propun un nou proces de acordare a canalelor, ilustrat în Fig. 31.

Pasul 1: Când se atinge un prag de congestie al celulei sau al unui grup de celule, la anumite intervale BTS va emite un mesaj PREAUTHENTICATION BEACON care conţine o valoare unică de 128 de biţi (nonce), cu viaţă redusă, similar cu cea folosită la autentificare (Fig. 32). Durata de viaţă limitată este determinată de BSC. La generarea fiecărei valori, BSC-ul va precalcula răspunsul aşteptat pentru fiecare cheie de utilizator înregistrată în baza sa de date (K_i) pentru ca ulterior verificările răspunsurilor dispozitivelor mobile să se efectueze într-un timp scurt.

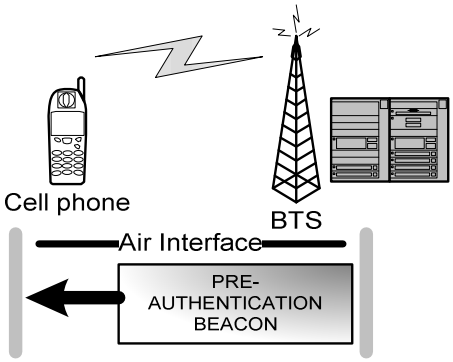


Fig. 32 Mesajul PREAUTHENTICATION BEACON

Valoarea de 128 de biţi este suficient de mare pentru a împiedica precalcularea unui spaţiu al cheilor statistic semnificativ, în special având în vedere că puterea de calcul a terminalelor mobile este limitată. Staţia mobilă memorează ultima valoare primită, fiind folosită pentru cereri ulterioare de alocare de canal, atât timp cât valoarea TTL (Time To Live) permite. Faza de preautentificare funcţionează asemănător cu autentificarea în sine. Răspunsul ar trebui însă scurtat pentru a scădea traficul de pe canalul cu acces aleatoriu, care altfel ar deveni congestionat iar acest lucru ar anula avantajul introdus de preautentificare. Propun ca răspunsul să fie redus la 16 biţi din cei 32, având în acest fel un spaţiu de 65536 de valori, suficiente pentru a evita potrivirile ocazionale în cazul în care atacatorul ar trimite o rafală de cereri false cu răspunsuri aleatoare.

Procesul este ilustrat în Fig. 33.

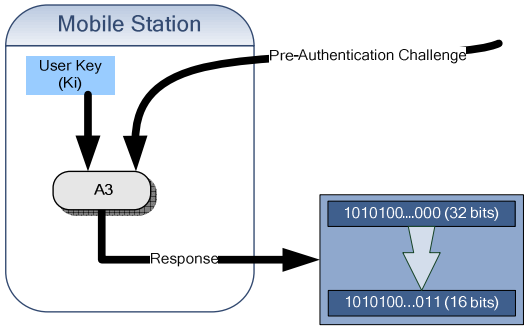


Fig. 33 Calcularea răspunsului de preautentificare

Mesajul CHANNEL REQUEST extins trimis prin canalul de acces aleator (RACH – Random Access Channel) trebuie să conţină atât motivul cererii resursei (ca în versiunea originală) cât şi răspunsul de preautentificare de 16 biţi.

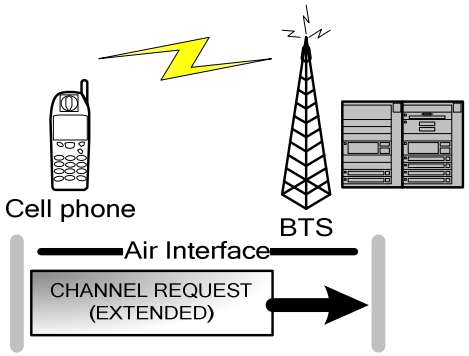


Fig. 34 Mesajul CHANNEL REQUEST (EXTENDED)

4.5.1 Impactul preautentificării asupra capacităţii de semnalizare

Schema de autentificare propusă necesită transportul unei cantităţi de 16 biţi pe canalul cu acces aleatoriu (RACH – Random Access Channel). Pe RACH, mobilul comunică prin aşa numitele rafale de acces (AB – Access Burst), având următoarea structură:

Tail Bits 8	Syncho Sequence 41	Encrypted Bits 36	Tail Bits 3	Guard Period 68,25

Informaţia utilă transportată în AB este de 36 de biţi din care primii 8 biţi sunt mesajul de acces (3 biţi tipul de acces, 5 biţi codul aleatoriu de culoare pentru a distinge între mai multe mobile care efectuează simultan o cerere) iar restul de 28 de biţi sunt folosiţi pentru coduri CRC şi de convoluţie.

Pentru implementarea preautentificării este astfel nevoie de un AB suplimentar care să transporte valoarea pe 16 biţi calculată de mobil urmată de un cod CRC pentru a asigura corectitudinea transmisiei. Legătura dintre cele două AB se poate face prin cei 5 biţi aleatorii de culoare la care se adaugă un număr de secvenţă pentru a putea discrimina componentele mesajului multiplu.

Având în vedere că un cadru are lungimea de 0,234 secunde iar un multicadru de tip RACH are 51 de cadre, avem aproximativ 218 accese RACH posibile pe secundă într-o singură celulă. Deoarece protocolul de stabilire a apelului în noua formă presupune două accese consecutive la canalul RACH, obţinem o înjumătăţire a capacităţii de semnalizare la 114 cereri pe secundă într-o celulă.

4.5.2 Avantaje şi limitări ale soluţiei propuse

Efectuarea preautentificării la fiecare cerere de acces la resurse la nivelul BSC-ului asigură continenţa resurselor. Deoarece canalul RACH pe care se fac accesele are capacitate limitată, schema de preautentificare este slăbită în mod deliberat prin eliminarea a jumătate dintre cei 32 de biţi rezultaţi. Un atacator care ar încerca să lanseze cereri pentru care ghiceşte valorile de răspuns ar avea o rată de succes de 2^{-16}. Având în vedere faptul că în sistemul GSM se pot efectua aproximativ 784000 de accese de tip burst pe RACH pe oră şi presupunând că un terminal mobil foloseşte toate aceste oportunităţi pentru a încerca ghicirea valorii de răspuns, vom avea în medie $784000 * 2^{-16} = \sim 12$ ghiciri corecte pe oră, cantitate neglijabilă care nu permite un atac de tip DoS.

Pentru a putea lansa cu succes un atac, atacatorul trebuie să aibă acces la un număr mare de chei de utilizator (K_i) valide şi înregistrate în reţea. Entitatea care are o listă cu astfel de chei este OMC-ul, deci atacatorul trebuie să folosească o combinaţie de mecanisme tehnice şi de inginerie socială precum şi de informaţii din interiorul reţelei pentru a obţine această listă şi a o folosi ulterior pentru precalcularea valorilor de răspuns ale mesajului CHANNEL_REQUEST (EXTENDED).

4.6 Concluzii

Disponibilitatea şi securitatea reţelelor GSM este un subiect actual, în condiţiile în care infrastructura de comunicaţie a societăţii se bazează în mare măsură pe reţelele mobile, mai ales cele GSM. Capitolul de faţă propune în premieră clasificarea vulnerabilităţilor din GSM după o metodă folosită în mod uzual în software. Fiecărei vulnerabilităţi i se atribuie note corespunzătoare următorilor

factori: potenţialul de distrugere, reproductibilitatea, exploatabilitatea, utilizatorii afectaţi, uşurinţa în descoperire. Media acestor note dă nivelul de gravitate al vulnerabilităţii, după care se poate efectua clasificarea. În urma clasificării, pe primul loc se situează atacurile de tip DoS, datorită relativei simplităţi cu care poate fi lansat şi mai ales datorită numărului mare de utilizatori afectaţi în timpul atacului.

S-a arătat de asemenea că atacurile DoS sunt posibile datorită faptului că reţeaua GSM se bazează pe onestitatea clienţilor. Resursele din domeniul radio sunt alocate fără nicio autentificare prealabilă, lucru ce duce la blocaje dacă un singur client face cereri în rafală cu scopul de a epuiza numărul de canale disponibile în celulă.

Contribuţiile aduse de acest capitol pot fi sumarizate după cum urmează:

- Vulnerabilităţile sistemului GSM au fost evaluate cu o metodă de clasificare numită DREAD. Rezultatul deloc surprinzător a fost că cea mai periculoasă ameninţare la adresa GSM este atacul DoS (denial of service).
- S-a arătat cauza pentru care sistemul GSM este vulnerabil la atacuri DoS, şi anume lipsa preautentificării dispozitivelor mobile care pot cere alocarea de resurse în mod repetat.
- S-a arătat de asemenea că un singur atacator este capabil de a dezactiva o întreagă celulă GSM şi în anumite cazuri chiar şi celulele adiacente.
- Deoarece nu sunt implicate costuri financiare (nu se efectuează nici un apel în sine), costul efectiv al lansării un atac devastator este zero din punctul de vedere al plăţilor către operatorul vizat.
- În scopul creşterii rezistenţei la atacuri DoS, s-a propus introducerea preautentificării terminalelor GSM în cazul iniţierii unui apel de către acestea. Preautentificarea impune introducerea în protocolul Channel Assignment a unui mesaj nou de tip baliză numit PREAUTHENTICATION BEACON care transmite terminalelor mobile o valoare challenge care va fi folosită ulterior ca preautentificare în protocolul de iniţiare al apelului.
- S-a arătat că impactul preautentificării pentru reţeaua GSM este înjumătăţirea capacităţii de semnalizare pe canalul RACH.

5. Contribuţii în domeniul sistemelor de distribuţie digitală a conţinutului

Multitudinea de dispozitive electronice portabile care ne înconjoară a atras implicit şi nevoia acestora de comunicare. De la simple mesaje scrise sau e-mail, aceste dispozitive sunt acum capabile de a reda conţinut digital audio şi video pe care îl obţin fie de la o sursă locală cum ar fi un card de memorie fie de la o sursă online de tip streaming sau download. Mereu în actualitate, problema drepturilor de autor este acum şi mai pregnantă. Dispozitivele electronice portabile sunt o piaţă gata de exploatat de către casele de discuri şi cele de film, cu condiţia ca veniturile din distribuţia de muzică şi film să nu fie afectate într-o măsură semnificativă de piraterie.

Consumatorii de entertainment digital mobil sunt în special tinerii. Odată cu lansarea unui album de muzică nou sau al unui film, utilizatorii vor provoca creşteri momentane ale valorilor traficului care în lipsa unei infrastructuri suficient dimensionate pot constata degradări ale serviciului consumat, cum ar fi timpi de aşteptare mari, viteze de transfer scăzute, etc. Dimensionarea corespunzătoare a infrastructurii trebuie astfel dublată de eficientizarea transportului de informaţii.

Drepturile de distribuţie pentru un conţinut digital sunt deţinute arareori de o singură entitate. De cele mai multe ori, conţinutul digital este rezultatul colaborării între diverse entităţi cum ar fi producători video, de sunet, text, interpretare, distribuţie, media, etc. Câştigul de pe urma comercializării produsului final se împarte între acestea într-un cadru contractual stabilit anterior. Spre deosebire de distribuţia clasică ce presupune vânzarea unui produs tangibil cum ar fi CD sau DVD, în cazul distribuţiei digitale nu există un control asupra volumului de vânzări exercitat direct de către toate entităţile implicate. Colaborarea în acest caz este de preferat să se producă la nivel interactiv, în sensul că toate părţile implicate să îşi dea acordul momentan pentru distribuţia de conţinut pentru un client.

Pe piaţă există deja un număr de sisteme de distribuţie digitală a conţinutului. În principal, acestea se concentrează pe argumentele tehnice privitoare la mecanismul de redare a conţinutului pe dispozitivele autorizate. Scalabilitatea unor astfel de sisteme este redusă şi nu poate face faţă unor vârfuri de trafic şi în plus nici nu se are în vedere colaborarea între părţile care partajează drepturile asupra unui conţinut digital.

În cadrul acestui capitol se va prezenta o arhitectură de distribuţie digitală a conţinutului care îmbunătăţeşte într-o manieră semnificativă scalabilitatea şi maniera de cooperare între deţinătorii de drepturi de autor. Arhitectura de la care s-a pornit este cea descrisă în [63].

5.1 Arhitectură scalabilă de distribuţie digitală a conţinutului

Marele neajuns al sistemelor de distribuţie digitală a conţinutului existente astăzi este că în general fac uz de criptografia asimetrică pentru protejarea conţinutului (direct, prin criptarea acestuia sau indirect, prin criptarea unei chei simetrice de sesiune), fapt ce limitează drastic scalabilitatea. Arhitectura propusă aici ameliorează acest neajuns prin gruparea clienţilor şi efectuarea operaţiilor costisitoare ca timp de un număr mai redus de ori, astfel încât scalabilitatea creşte în aceleaşi condiţii de putere de calcul.

Sistemul de distribuţie digitală a conţinutului propus are structura generală prezentată în Fig. 35.

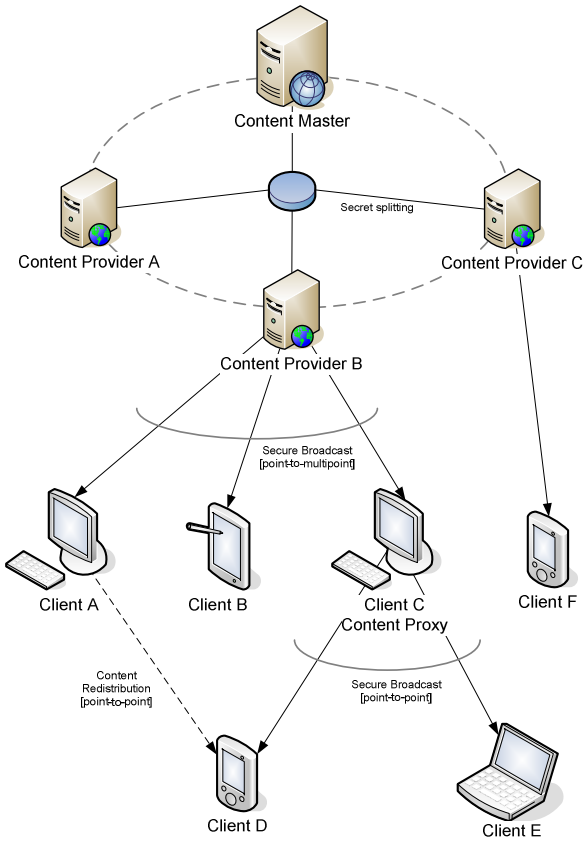


Fig. 35 Arhitectura sistemului scalabil de distribuţie a conţinutului

Actorii din cadrul sistemului sunt:

- **Content Provider (CP)** este o entitate care deţine drepturile totale sau parţiale asupra unui conţinut digital. Dacă drepturile deţinute sunt parţiale, pentru distribuţia conţinutului acesta va solicita aprobarea tuturor celorlalţi CP implicaţi şi care partajează drepturile de distribuţie a respectivului conţinut digital.
- **Content Master (CM)** este o entitate de încredere care reprezintă organizaţia şi care controlează activitatea tuturor CP şi este implicată indirect în procesul de distribuţie a conţinutului. Se poate considera că CM este autoritatea care supervizează procesul de aprobare şi arbitrare între CP şi generează cheile de sesiune pentru activităţile CP.
- **Clienţii** sunt dispozitive mobile sau programe software care au dreptul de a reda conţinutul multimedia recepţionat de la un CP şi opţional, ar putea deţine drepturile de redistribuţie. Pentru ca sistemul de drepturi să fie viabil, clienţii trebuie să îndeplinească o serie de condiţii specificate în [92]. Mai mult, fiecărui dispozitiv sau program software îi este alocată o pereche de chei (una publică iar alta privată) ce se foloseşte la schimbul de informaţii dintre CP şi un alt client. Procesul de redistribuţie ce are loc între clienţii A şi D (Fig. 35) este dictat de politicile DRM asociate conţinutului (ex. conţinutul poate fi redistribuit doar un anumit număr de ori, doar anumitor clienţi, etc.). Dacă politicile DRM nu sunt respectate de către client, CP-ul poate să-i revoce drepturile de distribuţie.
- **Content Proxy (CPx)** este o funcţie a unui client obişnuit care doreşte – gratuit sau contra unui avantaj – să-şi pună la dispoziţie puterea de calcul şi banda de comunicaţie pentru a ameliora încărcarea CM-ului. Orice client poate deveni content proxy dacă doreşte acest lucru, semnalizând intenţia unui CP. Deoarece conţinutul digital este criptat, CPx-ul nu va putea să redea conţinutul dacă acesta nu i se adresează în mod direct, dar va putea să trimită mai departe conţinutul destinatarilor legitimi. În Fig. 35, clientul C are rolul de content proxy, distribuind datele clienţilor D şi E folosind un broadcast securizat. Prin utilizarea proxy-urilor, CM îşi poate reduce drastic încărcarea deoarece odată procesate cererile de conţinut, distribuţia acestora este preluată de un CPx.

Procesul are două părţi distincte: **distribuţia conţinutului** (de la furnizorul de conţinut CP la clientul C_1) şi **redistribuţia conţinutului** (de la un client C_1 la altul C_2). Distribuţia poate avea loc fără a fi urmată de redistribuţie, însă reciproca nu este valabilă. Se introduc următoarele notaţii:

e_A/d_A – Perechea de chei public/privată a entităţii A.

$[D]e_A$ – Datele D criptate cu cheia publică a entităţii A.

$[D]d_A$ – Datele D semnate cu cheia publică a entităţii A.

$[D]_K$ – Datele D criptate cu cheia simetrica K.

h(D) – o funcţie de dispersie fără coliziuni aplicată datelor D

5.1.1 Distribuţia conţinutului

Furnizorul de conţinut (CP) distribuie conţinutul digital M şi drepturile asociate R, clienţilor (Ci) cu permisiunea celorlalţi furnizori (CP2, ..., CPn) şi sub supravegherea furnizorului master (MCP).

(1) C1, C2, …, Cn -> CP1: Solicitare conţinut

(2) C1, C2, …, Cn <-> CP1: Autentificare reciprocă

(3) CP1 <-> CP2, …, CPn, MCP: Acord asupra drepturilor de distribuţie

(4) C1, C2, …, Cn <-> CP1: Plată (pas opţional)

(5) CP1 -> C1, C2, …, Cn : [M]K, [K]eS, X, η, δ, Λ

Furnizorul de conţinut aşteaptă cereri de distribuţie de la clienţi şi le deserveşte la intervale prestabilite, spre exemplu o dată la câteva secunde. În acest timp se acumulează un număr de cereri pentru acelaşi conţinut digital, care sunt grupate în pasul (1). În pasul (2), furnizorul de conţinut şi clienţii se autentifică reciproc. Spre deosebire de arhitectura descrisă în [63], arhitectura de faţă nu prevede ca plata să se realizeze în acest stadiu întrucât furnizorul de conţinut (CP) care a primit cererea nu are autorizarea omologilor săi de a distribui conţinutul. În pasul (3), furnizorii de conţinut se pun de acord asupra autorizării cererii de distribuţie primite de CP şi în pasul (4) se efectuează plata.

În pasul (5), furnizorul de conţinut generează un lacăt X, criptează conţinutul cu o cheie simetrică K ce se foloseşte o singură dată, criptează K cu cheia de sesiune eS şi apoi trimite conţinutul criptat împreună cu lacătul X, drepturile η, metadata δ şi licenţa pentru conţinut Λ. Drepturile η reprezintă o cantitate care descrie cum se va trata conţinutul digital pe dispozitivele aprobate iar metadata δ sunt informaţiile asociate conţinutului (numele artistului, albumul, numele melodiei, rata de compresie, etc.). Licenţa de conţinut Λ se defineşte astfel:

$$\Lambda = [h(M, \eta, \delta, X)]_{dCP}$$

Scopul lui Λ este să certifice acordarea drepturilor η către client pentru conţinutul M. Drepturile η pot fi reprezentate folosind limbaje de autorizare si politici de acces cum ar fi XACML [94] şi XrML [95].

Arhitectura propusă prezintă două particularităţi: în pasul (3) se efectuează punerea de acord a CP asupra drepturilor de distribuţie prin tehnica împărţirii secretului [76][75] iar în pasul (5) se face un broadcast securizat pentru clienţii cărora le este destinat conţinutul, tehnică bazată pe lacătele securizate [25] folosind teorema chineză a restului [93]. Aceste aspecte se vor detalia în paragrafele care urmează.

5.1.1.1 Cooperarea furnizorilor de conţinut

Furnizorii de conţinut pot partaja drepturile de distribuţie, caz în care trebuie ca toţi să aprobe şi opţional să memoreze cererile clientului. Cooperarea se face prin partajarea unui secret şi participarea furnizorilor la reconstrucţia sa. Nici un furnizor nu poate obţine secretul fără cooperarea celorlalţi furnizori participanţi.

Considerând un mesaj secret M de lungime m şi un grup de n entităţi care partajează un secret, denominaţi P_1, P_2, ..., P_n, secretul poate fi împărţit între cei n după cum urmează:

1. Se generează n-1 numere aleatoare de lungime m, notate R_1, R_2, ... , R_{n-1}.
2. Mesajul M este criptat astfel: $S = M \otimes R_1 \otimes R_2 \otimes \ldots \otimes R_{n-1}$
3. Secretul S este distribuit lui P_1, R_1 este distribuit lui P_2, R_2 lui P_3 şi aşa mai departe până la R_{n-1} care este distribuit lui P_n.

Se observă că singura cale de a obţine M este prin aplicarea funcţiei XOR bucăţilor distribuite celor care deţin parte din secret. Aceştia nu trebuie să cunoască cine a primit S şi cine a primit şirurile aleatoare R_i. Acest lucru face sigură tehnica împărţirii secretului.

În cazul nostru, considerând că schema de distribuţie conţine un MCP şi un număr n de CP: CP_1, CP_2, ..., CP_n care partajează drepturile de distribuţie pentru conţinutul digital, tehnica funcţionează după cum urmează:

1. CP_i primeşte o cerere de la client.
2. Dacă CP_i acceptă cererea (adică dacă clientul este credibil, contul este pe balanţă pozitivă, a trecut verificările de bonitate, etc.), acesta va cere MCP-ului – care este autoritatea de încredere – să creeze şi distribuie un mesaj secret M pentru toţi furnizorii de conţinut (CP).

3. MCP generează un mesaj secret M, cunoscut numai de el, îl împarte în n bucăţi şi le împarte tuturor furnizorilor de conţinut (CP) prin utilizarea tehnicii de împărţire a secretului.
4. CP_i trimite cererea originală a clientului tuturor celorlalţi CP, spre informare şi o corectă luare de decizie.
5. Dacă un furnizor de conţinut este de acord cu conţinutul cererii clientului, acesta va pune la dispoziţia CP_i partea sa de secret.
6. Având răspunsurile celorlalţi deţinători de secrete, CP_i va reconstrui mesajul M, care va fi trimis lui MCP pentru validare. Mesajul rezultat M a fi valid dacă toţi furnizorii de conţinut şi-ai dat acordul pentru distribuţia conţinutului supus discuţiei.
7. MCP va proceda la verificarea mesajului primit ca răspuns cu mesajul original creat în pasul 3. Dacă validarea se produce cu succes, MCP va autoriza CPi să distribuie conţinutul digital clientului, altfel va instrui CP să respingă cererea clientului.

5.1.1.2 Broadcast securizat către clienţi

Abordarea tradiţională a distribuţiei de conţinut presupune ca pentru fiecare cerere de la un client, CP-ul să cripteze conţinutul folosind cheia publică a clientului care a făcut cererea [63]. Un număr de n clienţi care solicită conţinut digital de la CP va necesita n criptări, chiar dacă conţinutul este identic pentru toţi clienţii. Acest lucru înseamnă o încărcare inutilă pe serverul care deserveşte clienţii şi se traduce printr-o scalabilitate redusă.

Pentru a reduce încărcarea pe server, se propune ca furnizorul de conţinut (CP) care deserveşte cererile clienţilor să folosească o tehnică de distribuţie securizată pentru toţi clienţii care au cerut un anumit conţinut. În acest caz, conţinutul digital este criptat o singură dată folosind o cheie de sesiune cunoscută numai de CP, fiind trimis clientului împreună cu o cheie de decriptare. Pentru a se asigura că doar clienţii legitimi au acces la această cheie de decriptare, aceasta trebuie protejată cu un lacăt. Lacătul poate fi îndepărtat numai de clienţii legitimi şi este generat prin aplicarea unei tehnici numite teorema chineză a restului (CRT – Chinese Remainder Theorem) [25]:

Tehnica funcţionează astfel: Se dă un grup de n clienţi notat C, format din C_1, C_2, ..., C_n, fiecare având o cheie privată e_i, respectiv una publică d_i. Se dă de asemenea un set de n numere pozitive întregi N_1, N_2, ..., N_n cu proprietatea că sunt numere prime între ele şi sunt cunoscute public în sistem. Din grupul C, un subset de k clienţi ($k \geq 2$) solicită acelaşi conţinut digital de la furnizorul de conţinut (CP). Dacă în continuare considerăm un set de k întregi pozitivi R_1, R_2, ... R_n, teorema chineză a restului afirmă că sistemul de congruenţe:

$$\begin{cases} X \equiv R_1 (\mathrm{mod}\, N_1) \\ X \equiv R_2 (\mathrm{mod}\, N_2) \\ \cdots \\ X \equiv R_k (\mathrm{mod}\, N_k) \end{cases}$$

Ec. (8)

are o soluţie comună X, dată de ecuaţia:

$$X = (\sum_{i=1}^{k} \frac{L}{N_i} \cdot R_i \cdot f_i) \bmod L$$

Ec. (9)

unde:

$$1 \equiv f_i \cdot \frac{L}{N_i} \bmod N_i$$

Ec. (10)

Cu L definit ca $\prod_{i=i}^{k} N_i$

În schema propusă, furnizorul de conţinut va genera o pereche de chei de sesiune e_s şi d_s, şi va cripta conţinutul digital o singură dată, folosind e_s. Furnizorul va genera de asemenea numerele R_1, R_2, ... R_n prin codificarea cheii de decriptare d_s cu cheia e_i a fiecărui client, după cum urmează: $R_i = Enc_{e_i}(d_S)$

Lacătul X se obţine rezolvând sistemul de congruenţe şi este trimis clienţilor împreună cu conţinutul digital criptat. Fiecare client *i* din grupul care a solicitat conţinutul digital poate obţine R_i din lacătul X, iar prin decriptarea cu cheia privata d_i poate obţine d_S. Odată ce clientul are cantitatea d_S, acesta poate decripta conţinutul digital şi îl poate utiliza în conformitate cu politica DRM asociată.

Din tehnica descrisă anterior se observă că lacătul nu poate fi deschis de către clienţi nelegitimi. Chiar dacă un client rău intenţionat obţine restul R_i, acest nu poate extrage d_S deoarece nu posedă cheia privată d_i a clientului legitim *i*.

5.1.2 Retransmisia conţinutului

Prin faptul că un client nu poate decodifica decât conţinutul care îi este direct adresat, sistemul de distribuţie permite o degrevare suplimentară a serverului la momentul transmisiei de fapt a datelor. Conţinutul digital care trebuie livrat la un moment dat în mod normal s-ar plasa într-o coadă care ar deservi clienţii care se conectează secvenţial sau simultan în limita lăţimii benzii de comunicaţie. În cazul arhitecturii propuse însă este posibil ca mai întâi să se trimită conţinutul digital de la un moment dat către clienţi mai apropiaţi geografic de destinatari prin intermediul aşa numiţilor clienţi proxy. Aceşti clienţi participă indirect la procesul de distribuţie, strict în sensul transportului de informaţie.

Procesul de retransmisie a conţinutului este arătat în Fig. 36. Presupunem că furnizorul de conţinut a autorizat toţi clienţii şi a primit acceptul altor furnizori de a trimite conţinutul.

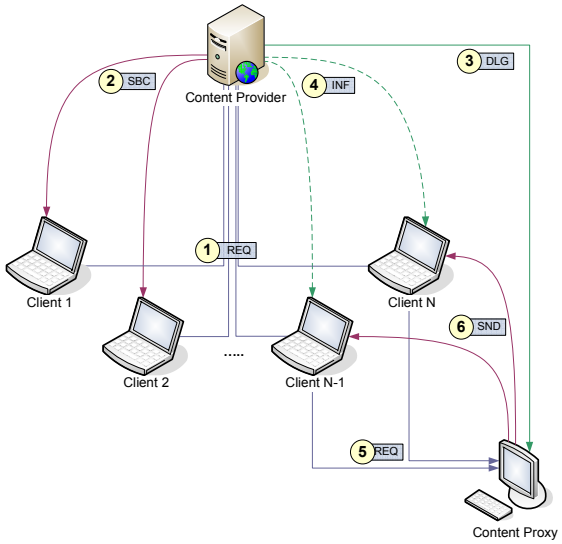


Fig. 36 Retransmisia conţinutului prin clienţi proxy

1. În urma cererii clienţilor (marcate cu REQ), CP creează conţinutul digital într-un mesaj de ieşire (după tehnica descrisă anterior) şi îl trimite celor N clienţi care l-au solicitat, folosind tehnica difuzării securizate cu lacăt. Când clientul primeşte mesajul, acesta va trimite o înştiinţare (ACK) către CP.
2. CP verifică dacă toţi cei N clienţi au primit mesajul. Dacă sunt clienţi care au răspuns cu NAK (în cazul nostru clienţii N-1 şi N) sau dacă serverul este încărcat, CP va trimite mesajul original la unul sau mai mulţi clienţi proxy înregistraţi şi va delega sarcina de a distribui mesajul (DLG).

3. CP informează clienţii care nu au recepţionat mesajul de existenţa unui client proxy care este în posesia unei copii a conţinutului digital. Din acest moment, CP este degrevat de sarcina de a distribui conţinutul.
4. Clienţii notificaţi de CP în pasul 4 vor solicita conţinutul digital direct de la proxy, care deţine o copie codificată a conţinutului şi nu va fi capabil să o decodifice decât dacă se află pe lista destinatarilor finali.
5. Proxy-ul va înainta mesajul codificat fiecărui client pentru care a primit notificare.

5.1.3 Redistribuţia conţinutului

Drepturile η acordate iniţial de către furnizorul de conţinut pot permite clientului C_1 să redistribuie conţinutul către un alt client C_2, urmând protocolul următor [63]:

(1) C2 -> C1: Solicitarea conţinutului

(2) C2 <-> C1: Autentificare reciprocă

(3) C1 -> C2: [M]K', [K']eC2, η, η', δ, Λ, Λ'

(4) C2 <-> C1: verificarea cantităţii δ, plata (opţională)

(5) C2 -> C1: φ

Clientul C_2 porneşte tranzacţia în pasul (1) prin solicitarea lui C_1 a unui conţinut digital anume. În pasul (2), cele două părţi se autentifică reciproc folosindu-şi perechile de chei publice/private. Dacă autentificarea are succes, C_1 decriptează conţinutul digital solicitat, generează o cheie temporară simetrică K' şi criptează conţinutul cu această cheie. Cheia K' este apoi criptată cu cheia public a clientului C_2. C_1 trimite apoi către C_2 noul conţinut criptat, cheia de sesiune K' criptată cu cheia publică a lui C_2, drepturile iniţiale η şi noile drepturi η'. De asemenea, C_1 trimite licenţa iniţială Λ şi nouă licenţă Λ', definită astfel:

$$\Lambda' = [h(e1, e2, M, \eta', \delta, X)]_{dC1}$$

În pasul (4), C_2 verifică semnătura lui C_1 pe noua licenţă Λ' şi validează η şi M folosind licenţa Λ originală. C_2 se asigură de asemenea că licenţa η' poate fi derivată din η şi verifică de asemenea δ faţă de tipul de conţinut distribuit. Dacă toate verificările trec cu succes, C_2 aprobă tranzacţia din pasul (5), trimiţând lui C_1 o înştiinţare φ, definită ca:

$$\varphi = [h(eC1, eCP, [M]K', \delta, \eta']_{dC2}$$

Înştiinţarea φ reprezintă recunoaşterea lui C_2 că a primit conţinutul M cu drepturile η'.

5.1.4 Riscuri potenţiale

Arhitectura propusă introduce un număr de ameninţări, unele fiind comune cu arhitecturi DRM existente, altele fiind noi. Una dintre ameninţările cele mai răspândite este modificarea neautorizată a echipamentelor sau a modulelor protejate din interiorul acestora. Rezistenţa la modificări este dificil de obţinut [3][47], se poate deci spune că securitatea este eficace împotriva atacatorilor cu excepţia celor mai motivaţi.

Tehnicile criptografice sunt arareori ţinta unor atacuri întrucât atacatorii aleg puncte mai vulnerabile ale unui sistem, cum ar fi sistemul de redistribuţie şi colaborare între părţi. Se pot distinge următoarele tipuri atacuri:

- **Înlocuirea conţinutului** în timpul procesului de redistribuţie. Acest lucru se poate întâmpla când un client recepţionează o valoare mai mică decât cea comandată, sau clientul proxy înlocuieşte conţinutul înainte de a fi distribuit mai departe.
- **Copiile de siguranţă ale cheilor pe medii nesecurizate** poate duce la compromiterea cheilor deoarece acestea părăsesc mediul securizat oferit de modulul criptografic.
- **Dispozitivele compromise** sunt capabile de a îndepărta mecanismul de securitate care protejează conţinutul digital, putând astfel să distribuie conţinutul în mod ilegal. Detectarea şi izolarea dispozitivelor compromise este esenţială pentru sănătatea unui sistem DRM.
- **Compromiterea unui singur provider** duce la căderea totală a sistemului de distribuţie. Dacă un singur furnizor de conţinut are un comportament care nu este în acord cu sistemul, tehnic acesta nu va putea fi exclus uşor.
- Spre deosebire de alte sisteme care efectuează operaţii costisitoare (criptografie asimetrică), arhitectura pe care am prezentat-o este mai puţin susceptibilă la **atacuri denial-of-service**, întrucât sarcina de distribuţie propriu-zisă este delegată clienţilor proxy. Totuşi, atacurile nu pot fi prevenite în totalitate, dar există căi de ameliorare a efectelor [11][12].

5.1.5 Avantajele arhitecturii

Principalul avantaj al arhitecturii propuse este că scalează foarte bine în diverse scenarii. Când un anume conţinut digital este foarte cerut de către clienţi (spre exemplu un album nou de succes, un trailer al unui film sau chiar un film în sine), se poate întâmpla ca foarte mulţi dintre aceştia să ceară acest conţinut simultan şi într-un timp foarte scurt. În mod normal, furnizorul ar trebui să depună un efort susţinut pentru satisfacerea cererilor, efort ce presupune putere de calcul şi lăţime de bandă pentru comunicaţie. In acest caz însă, furnizorul de conţinut poate deservi cererile la un anumit interval (spre exemplu de ordinul secundelor) pentru a

se acumula un număr de cereri, apoi cu o singură operaţie de criptare serverul va putea deservi toţi clienţii respectivi, fapt ce duce la o posibilitate importantă de scalare a soluţiei. În plus, operaţiile de transfer de date pot fi preluate de clienţi cu rol de proxy, aceştia neputând reda conţinutul dacă nu le este adresat.

5.2 Implementare

Prototipul sistemului de distribuţie a conţinutului a fost implementat în limbajul C# pe platforma .NET Framework 3.5 sub numele de SCADDIC. Pentru a reduce din complexitatea prototipului s-au omis funcţionalităţile referitoare la confortul şi securitatea părţilor, dezvoltarea concentrându-se pe partea de concept şi demonstrare a fezabilităţii sistemului teoretic. Prototipul păstrează proporţiile diverselor sarcini relevante pentru evaluarea performanţei.

Cele trei secţiuni principale ale proiectului sunt:

- Structuri de date pentru descrierea conţinutului digital
- Clase pentru modelare diverselor entităţi care manipulează conţinutului
- Bibliotecă de operaţii matematice şi criptografice

Pentru a simplifica infrastructura de comunicaţie între entităţile care compun sistemul, s-a ales ca acestea să funcţioneze pe aceeaşi maşină. Comunicarea s-a putut realiza astfel prin simple apeluri de funcţii şi a fost posibil ca biblioteca de operaţii matematice şi criptografice să fie partajată între entităţi.

5.2.1 Structuri de date pentru descrierea conţinutului

Structurile de date pentru descrierea conţinutului au rolul de a clasifica conţinutul digital şi a datele auxiliare (metadata). Fiecare conţinut este identificat în mod unic de un cod, având date auxiliare asociate care descriu drepturile de redare şi descriptori precum titlu, autor, anul lansării, rata fluxului (bit rate), codec-ul folosit, etc. De asemenea, pentru a evita modificarea conţinutului în tranzit, este prezentă şi o sumă de control.

Clasa **ContentItem** conţine informaţia digitală a conţinutului, o sumă de control, precum şi datele auxiliare şi drepturile de distribuţie.

Clasa **ContentMetadata** conţine o listă de câmpuri cu valori asociate care au înţeles într-un anumit context dat, spre exemplu lista cu drepturi de redistribuţie, informaţii despre conţinutul digital (titlu, autor, bit-rate, etc.).

5.2.2 Entităţi care manipulează conţinutul

Conform design-ului sistemului de distribuţie descris, entităţile care manipulează conţinutul sunt:

Client – un computer sau un dispozitiv mobil care solicită, consumă sau redistribuie conţinutul digital, în concordanţă cu licenţele şi drepturile asociate. Acesta este implementat în clasa **ContentClient.**

```
namespace SCADDICCore
{
    /// <summary>
    /// Holds an association of the content item and its lock (we only need this locally, within the
client)
    /// </summary>
    public class LocalContentItem
    {
        #region Public variables
        /// <summary>
        /// The content item
        /// </summary>
        public ContentItem contentitem;
        /// <summary>
        /// Lock (X) associated to the content
        /// </summary>
        public BigInteger LockX;
        #endregion

        #region Public Constructor
        public LocalContentItem(ContentItem contentitem, BigInteger LockX)
        {
            this.contentitem = contentitem;
            this.LockX = LockX;
        }
        #endregion
    }

    /// <summary>
    /// The Content Client is the requester and the beneficiary of a digital content.
    /// </summary>
    public class ContentClient
    {
        #region Local Variables
        /// <summary>
        /// Content client name (to be distinguished among peers)
        /// </summary>
        public string Name
        {
            get;
            set;
        }

        /// <summary>
        /// A public number given by the Content Master at setup time
        /// </summary>
        public BigInteger N
        {
            get;
            set;
        }

        /// <summary>
        /// Asymmetric key cryptography provider
        /// </summary>
        protected RSACryptoServiceProvider RSA;

        /// <summary>
        /// ASCII Encoder/Decoder
        /// </summary>
        protected ASCIIEncoding ae;

        /// <summary>
        /// Content items stored locally on the client. This resembles the internal memory of a device in
real-world.
        /// </summary>
```

```
        protected SortedList<int, LocalContentItem> LocalContentItems;

        #endregion

        /// <summary>
        /// Initializes a new client
        /// </summary>
        /// <param name="Name">Name of the content client (to be distinguished among peers)</param>
        public ContentClient(string Name)
        {
            this.Name = Name;
            // Generate a large prime number that acts as N in the Chinese Remainder Theorem
            N = BigInteger.genPseudoPrime(1024, 1, new Random());
            ae = new ASCIIEncoding();
            RSA = new RSACryptoServiceProvider();
            LocalContentItems = new SortedList<int, LocalContentItem>();
        }

        #region Public Methods
        /// <summary>
        /// The client receives digital content via this method.
        /// </summary>
        /// <param name="LockX">The cryptographic lock X, from which the client will extract the session
key used to encrypt the content.</param>
        /// <param name="content">The received content.</param>
        /// <param name="License">License (lambda) from the content.</param>
        /// <param name="ContentMastePublicKey">The public key of the content master.</param>
        public void ReceiveContentFromMaster(ContentItem content, BigInteger LockX, ContentMetadata
License, RSAParameters ContentMastePublicKey)
        {
            Debug.WriteLine("Client " + Name + " received content from master.");
            // Compute R from the lock X and the publicly known value N (established at setup time)
            BigInteger R = LockX % N;
            string OriginalSessionKey = string.Empty;
            try
            {
                // Decode R with private key and get the session encryption key
                OriginalSessionKey = ae.GetString(RSA.Decrypt(R.getBytes(), false));
            }
            catch (Exception)
            {
                // Invalid R, the content is not for this client
            }

            // We now have both the encrypted file and the decryption key
            if (OriginalSessionKey == string.Empty)
            {
                Debug.WriteLine("The key is not for client " + Name + ", therefore it cannot decrypt the
content.");
            }
            else
            {
                // Verify signature with the public key of the content master
                string OriginalHash = License["LicenseHash"];
                byte[] Signature = Convert.FromBase64String(License["Signature"]);
                RSACryptoServiceProvider verifRSA = new RSACryptoServiceProvider();
                verifRSA.ImportParameters(ContentMastePublicKey);
                bool Verified = verifRSA.VerifyData(ae.GetBytes(OriginalHash), "MD5", Signature);

                if (!Verified) throw new Exception("Invalid content received");

                Debug.WriteLine("Decrypting content with key: " + OriginalSessionKey);
                // Decrypt content and analyze metadata, rights, license
                ContentCrypto.DecryptContent(content, OriginalSessionKey);

                // Generate content license hash (so we can check it whether it is genuine)
                // The hash will be compared against the one received from the master
                // Also, the signature will be verified
                string LicenseHash = Licensing.GenerateContentLicenseHash(content, LockX);
                if (LicenseHash != License["LicenseHash"])
                {
                    throw new Exception("License of the content is not valid.");
                }

                // Store received content (delete content first if we have it already in list)
                if (LocalContentItems.ContainsKey(content.ContentItemID))
                    LocalContentItems.Remove(content.ContentItemID);
                LocalContentItems.Add(content.ContentItemID, new LocalContentItem(content, LockX));
                // Consume content

                Debug.WriteLine("Content can be consumed.");
                Debug.WriteLine("Received metadata:");
```

```
                foreach (string field in content.MetaData.FieldNames)
                {
                    Debug.WriteLine(string.Format("Field: {0}, Value: {1}", field,
content.MetaData[field]));
                }

                Debug.WriteLine("Received rights:");
                foreach (string field in content.Rights.FieldNames)
                {
                    Debug.WriteLine(string.Format("Field: {0}, Value: {1}", field, content.Rights[field]));
                }

            }
        }

      /// <summary>
      /// Redistributes content to a peer client, should the rights allow that.
      /// </summary>
      /// <param name="TargetClient">Client to which the content will be sent.</param>
      /// <param name="LocalContentID">ID of the content subject to redistribution (in decrypted
state).</param>
        public void SendContentToPeer(ContentClient TargetClient, int LocalContentID)
        {
            Debug.WriteLine("Client " + Name + " redistributes content to client " + TargetClient.Name +
".");

            // TODO: Create the new set of rights from the original set
            // As of now, the original rights are just as the original ones, but a real implementation
            // would need to alter the rights in some way, ex. to decrease copy count.
            // If the new rights prevent the further distribution of the content, bail out.
            ContentItem OriginalContent = LocalContentItems[LocalContentID].contentitem;

            ContentMetadata NewRights = OriginalContent.Rights;

            // Create a copy of the current content so it does not interfere with our local copy
            ContentItem Content = new ContentItem(OriginalContent);
            // Compute the new content license
            string LicenseHash = Licensing.GenerateRedistributedContentLicense(Content,
LocalContentItems[LocalContentID].LockX, this.PublicKey, TargetClient.PublicKey, NewRights);
            // Digitally sign hash
            byte[] Signature = RSA.SignData(ae.GetBytes(LicenseHash), "MD5");

            ContentMetadata NewLicense = new ContentMetadata();
            NewLicense["LicenseHash"] = LicenseHash;
            NewLicense["Signature"] = Convert.ToBase64String(Signature);

            // Generate a new random key and encrypt the content with this one
            string SessionKey = RandomPassword.Generate(8);
            ContentCrypto.EncryptContent(Content, SessionKey);

            // Encrypt session key with the public key of the target client (receiver)
            RSACryptoServiceProvider targetrsa = new RSACryptoServiceProvider();
            targetrsa.ImportParameters(TargetClient.PublicKey);
            byte[] EncryptedSessionKey = targetrsa.Encrypt(ae.GetBytes(SessionKey), false);

            // Send the content to the client
            //TargetClient.ReceiveContentFromPeer(Content, EncryptedSessionKey, NewRights, NewLicense,
this);
        }

        /// <summary>
        /// Receives content from a peer client
        /// </summary>
        /// <param name="Content">Content received.</param>
        /// <param name="EncryptedKey">Encrypted key used to protect the content. The key can be retrieved
using the private RSA key.</param>
        /// <param name="NewRights">Redistribution rights for the current content.</param>
        /// <param name="NewLicense">Redistribution license for the current content.</param>
        /// <param name="From">Peer from which the content was received.</param>
        public void ReceiveContentFromPeer(ContentItem Content, byte[] EncryptedKey, ContentMetadata
NewRights, ContentMetadata NewLicense, ContentClient From)
        {
            // Verify license validity
            string LicenseHash = NewLicense["LicenseHash"];
            byte[] Signature = Convert.FromBase64String(NewLicense["Signature"]);
            RSACryptoServiceProvider verifRSA = new RSACryptoServiceProvider();
            verifRSA.ImportParameters(From.PublicKey);
            bool Verified = verifRSA.VerifyData(ae.GetBytes(LicenseHash), "MD5", Signature);

            if (!Verified) throw new Exception("Invalid content received");
            // TODO: Also compare hash of the license, not just the signature
```

```
            // Content received from a peer
            byte[] DecryptedKey = RSA.Decrypt(EncryptedKey, false);
            string ContentKey = ae.GetString(DecryptedKey);
            // Decrypt content
            ContentCrypto.DecryptContent(Content, ContentKey);

            Debug.WriteLine("Redistributed content can be consumed by client " + Name + ".");
        }

        /// <summary>
        /// Gets the public key of this content client
        /// </summary>
        /// <returns>The public key</returns>
        public RSAParameters PublicKey
        {
            get
            {
                return RSA.ExportParameters(false);
            }
        }

        #endregion
    }
}
```

Content Provider – Entitatea care deţine copyright-ul asupra conţinutului ce se distribuie. Acesta poate funcţiona în conjuncţie cu alţi provideri cu care partajează drepturile. Implementarea este în clasa **ContentProvider**.

```
namespace SCADDICCore
{
    /// <summary>
    /// The ContentProvider is responsible for granting/denying individual access to content
    /// by voting through secret splitting
    /// </summary>
    public class ContentProvider
    {
        #region Local Variables
        /// <summary>
        /// Content provider name (to be distinguished among peers)
        /// </summary>
        protected string Name;

        /// <summary>
        /// The list holds the secrets used to decript the content with a given ID. Each secret
        /// is used in conjunction with the secrets of other content providers in the secret splitting
technique.
        /// </summary>
        protected SortedList<int, string> ContentKeyPieces;

        #endregion

        #region Public Constructors
        /// <summary>
        /// Initializes a new content provider
        /// </summary>
        /// <param name="Name">Name of the content provider (to be distinguished among peers)</param>
        public ContentProvider(string Name)
        {
            ContentKeyPieces = new SortedList<int, string>();
            this.Name = Name;
        }

        #endregion

        #region Public Methods
        /// <summary>
        /// Receive a fragment of the key used to encrypt the content identified by ContentID. The rest of
the fragments
        /// are stored by peer Content Providers. All CPs will participate in the Secret Splitting schema
when requested.
        /// </summary>
        /// <param name="ContentID">Content identification</param>
        /// <param name="ContentKey">Key</param>
        public void SetContentKey(int ContentID, string ContentKey)
        {
```

```
            ContentKeyPieces.Add(ContentID, ContentKey);
            Debug.WriteLine("CP [" + Name + "] received key " + ContentKey + " for content ID " + ContentID
+ ".");
        }

        /// <summary>
        /// This is the heart of the authorization mechanism. The content provider checks whether ALL
specified
        /// clients are to be granted access to the content with ContentID.
        /// </summary>
        /// <param name="ContentID">Content ID to which access is requested.</param>
        /// <param name="ContentClients">List of clients that request authorization for the
content.</param>
        /// <returns>In case all clients are allowed acces, the provider will proceed with returning its
own
        /// secret key to participate in the secret sharing schema. If at least one client is not allowed,
the provider will return an empty string.</returns>
        public string AuthorizeContentDistribution(int ContentID, List<string> RequestingClients)
        {
            // TODO: As of now, the voting mechanism is not implemented. The content provider blindly
            // provides its key to the Content Master, without asking for reason or other details.
            string ContentKey = string.Empty;
            try
            {
                ContentKey = ContentKeyPieces[ContentID];
            }
            catch (Exception) { }

            Debug.WriteLine("CP [" + Name + "] issued key " + ContentKey + " for content ID " + ContentID +
".");

            return ContentKey;
        }
        #endregion

    }
}
```

Content Master – Este entitatea de încredere în sistemul de distribuţie. Cererile care vin de la clienţi sunt centralizate şi deservite aici, cu aprobarea furnizorilor de conţinut, care pot restricţiona accesul unui client în particular conform propriei voinţe. Implementarea este în clasa **ContentMaster**.

```
namespace SCADDICCore
{
    /// <summary>
    /// Summary for Content Master
    /// </summary>
    public class ContentMaster
    {
        /// <summary>
        /// Default secret length.
        /// </summary>
        protected const int PASSWORD_LENGTH = 8;

        #region Local Variables
        /// <summary>
        /// Content providers registered with the Content Master.
        /// </summary>
        protected List<ContentProvider> RegisteredContentProviders;
        /// <summary>
        /// Content items registered with the Content Master.
        /// </summary>
        protected SortedList<int, ContentItem> RegisteredContentItems;
        /// <summary>
        /// Hashes for the clear content of the registered items (so we can check the correct decryption
when give a certain password).
        /// </summary>
        protected SortedList<int, string> RegisteredContentHashes;
        /// <summary>
        /// Clients registered with the Content Master.
        /// </summary>
        protected SortedList<string, ContentClient> RegisteredContentClients;
        /// <summary>
        /// Content identifier. Increases with each new registered content.
        /// </summary>
```

```
        protected static int ContentID = 1;

        /// <summary>
        /// The RSA object of the content master. This is primarily used to sign content license (lambda)
        /// </summary>
        protected RSACryptoServiceProvider ContentMasterRSA;

        #endregion

        #region Public Constructors
        public ContentMaster()
        {
            RegisteredContentProviders = new List<ContentProvider>();
            RegisteredContentItems = new SortedList<int, ContentItem>();
            RegisteredContentClients = new SortedList<string, ContentClient>();
            RegisteredContentHashes = new SortedList<int, string>();
            ContentMasterRSA = new RSACryptoServiceProvider();
        }
        #endregion

        #region Public Methods

        /// <summary>
        /// Registers a new CP with the Content Master
        /// </summary>
        /// <param name="cp">Content Provider to register</param>
        public void RegisterContentProvider(ContentProvider newcp)
        {
            RegisteredContentProviders.Add(newcp);
        }

        /// <summary>
        /// Registers a new client with the Content Master
        /// </summary>
        /// <param name="cc">Client to register</param>
        public void RegisterClient(ContentClient cc)
        {
            RegisteredContentClients.Add(cc.Name, cc);
        }

        /// <summary>
        /// Retrieves a specified content client
        /// </summary>
        /// <param name="Name">Content client name</param>
        /// <returns>Content client information</returns>
        public ContentClient GetContentClient(string Name)
        {
            return RegisteredContentClients[Name];
        }

        /// <summary>
        /// Registers a media file for use in the SCADDIC system:
        /// 1. Generates a random password (the secret that will later be used in the secret splitting
technique)
        /// 2. Encrypts the file with the random password
        /// 3. Splits the password into equal pieces that will be fed to the content providers
        /// 4. Destroys the password so it can no longer be retrieved by an attacker
        /// </summary>
        /// <param name="ContentFileName">File containing the digital content to be registered</param>
        /// <param name="metadata">Metadata of the content being registered (e.g. bit rate, author name,
etc.)</param>
        /// <param name="rights">Rights associated to the content being registered (e.g. NO COPIES,
etc.)</param>
        /// <returns>The ID of the content being registered.</returns>
        public int RegisterContentItem(string ContentFileName, ContentMetadata Metadata, ContentMetadata
Rights)
        {
            // Generate a random password
            string Password = RandomPassword.Generate(PASSWORD_LENGTH);
            // Debug.WriteLine("Random password at content registration: " + Password);

            // Let N be the number of currently registered CPs. Generate N-1 secrets of length
PASSWORD_LENGTH.
            int N = RegisteredContentProviders.Count - 1;
            // If no content providers are registered, throw an exception
            if (N == -1) throw new Exception("No content providers are registered. Please register at least
on CP before you can register a content item.");

            string[] R = new string[N];
            for (int i = 0; i < R.Length; i++)
            {
                R[i] = RandomPassword.Generate(PASSWORD_LENGTH);
```

```
            }
            // Calculate S defined as Password XOR R1 XOR ... XOR R n-1
            string S = Password;
            for (int i = 0; i < N; i++)
            {
                S = S.XOR(R[i]);
            }

            // Now we have S, R1, R2,..., Rn-1. We have to distribute these values to CPs.

            // Get a new ID for the current content. IDs will be sequentially incremented with each new
request.
            int ContentID = GetNewContentID();
            // Distribute S to the first content provider
            RegisteredContentProviders[0].SetContentKey(ContentID, S);
            // Distribute subsequent parts of the secret (R1, R2,..., Rn-1) to the rest of the CPs.
            for (int i = 1; i <= N; i++)
            {
                RegisteredContentProviders[i].SetContentKey(ContentID, R[i-1]);
            }

            // Create a new content item
            ContentItem newci = new ContentItem(ContentID, File.ReadAllBytes(ContentFileName), Metadata,
Rights);
            // Add hash to list (while the content is still unencrypted)
            RegisteredContentHashes.Add(ContentID, newci.ContentHash);
            // Encrypt the new content
            ContentCrypto.EncryptContent(newci, Password);
            // Add content to list (so we know it later)
            RegisteredContentItems.Add(ContentID, newci);
            // Return the ID of the content being registered
            return ContentID;
        }

        /// <summary>
        /// Deletes the specified content item from the master store
        /// </summary>
        /// <param name="ContentID">ID of the content to destroy</param>
        public void UnregisterContentItem(int ContentID)
        {
            RegisteredContentItems[ContentID] = null;
        }

        /// <summary>
        /// Implements the secret splitting technique.
        /// </summary>
        /// <param name="ContentID">Content ID being voted by the content providers.</param>
        /// <param name="ContentClients">Clients requesting the content.</param>
        /// <returns>TRUE if the voting allows the distribution of the content item. Should this value be
false, the content of the DecryptedFile out variable should be discarded.</returns>
        protected ContentItem CheckContentProviderAuthorization(int ContentID, List<string>
RequestingClients)
        {
            if (RegisteredContentProviders.Count == 0) throw new Exception("Please register at least one
content provider.");

            string S = RegisteredContentProviders[0].AuthorizeContentDistribution(ContentID,
RequestingClients);
            for (int i = 1; i < RegisteredContentProviders.Count; i++)
            {
                S = S.XOR(RegisteredContentProviders[i].AuthorizeContentDistribution(ContentID,
RequestingClients));
            }
            // S should now hold the original secret
            Debug.WriteLine("Decrypted password (after voting): " + S);


            // Let's see what we know about the current content item
            ContentItem original = RegisteredContentItems[ContentID];
            // In order not to make changes into the registered list of content items, we will work on a
copy
            ContentItem decrypteditem = new ContentItem(original);
            // Decrypt content (using S as a password)
            ContentCrypto.DecryptContent(decrypteditem, S);

            // Verify checksum
            if (decrypteditem.ContentHash == RegisteredContentHashes[ContentID])
            {
                // We're ok, the voting allows us to distribute the content
                return decrypteditem;
            }
            else
```

```
            {
                return null;
            }
        }

        /// <summary>
        /// Performs a secure broadcast to the specified clients
        /// </summary>
        /// <param name="ContentID">ID of the content that will be broadcast</param>
        /// <param name="RequestingClients">List of clients (names) that will receive the content</param>
        public void BroadcastContent(int ContentID, List<string> RequestingClients)
        {
            // Before we can do anything, we must first check whether all content providers agree to grant
the content
            // to the all clients we have.
            #region Check Authorization
            DateTime start = DateTime.Now;
            ContentItem BroadcastMessage = CheckContentProviderAuthorization(ContentID, RequestingClients);
            if (BroadcastMessage == null) throw new Exception("Content with ID " + ContentID + " hasn't
been authorized for all clients.");
            #endregion

            // Generate a random session password that will encrypt the content. The password will be
protected by the lock
            // X which will be opened by authorized clients only.
            string SessionKey = RandomPassword.Generate(8);

            // Calculate lock X
            #region Calculate lock X
            BigInteger X = ComputeLockX(SessionKey, RequestingClients);
            #endregion

            // Compute content license (lambda)
            string LicenseHash = Licensing.GenerateContentLicenseHash(BroadcastMessage, X);

            // Digitally sign license hash
            ASCIIEncoding ae = new ASCIIEncoding();
            byte[] Signature = ContentMasterRSA.SignData(ae.GetBytes(LicenseHash), "MD5");
            ContentMetadata License = new ContentMetadata();
            License["LicenseHash"] = LicenseHash;
            License["Signature"] = Convert.ToBase64String(Signature);

            // Encrypt content with session password
            ContentCrypto.EncryptContent(BroadcastMessage, SessionKey);
            DateTime end = DateTime.Now;
            Debug.WriteLine("Content generated for broadcast in: " + (end - start).TotalMilliseconds + "
ms. ======================================================");

            // Send (broadcast) everything to each client (i.e. the encrypted content and the X)
            foreach (ContentClient cc in RegisteredContentClients.Values)
            {
                // TODO: Uncomment this line when not benchmarking
                cc.ReceiveContentFromMaster(new ContentItem(BroadcastMessage), X, License,
ContentMasterRSA.ExportParameters(false));
            }
        }
        #endregion

        #region Helper Methods
        /// <summary>
        /// Computes the lock X for the current list of clients requesting
        /// </summary>
        /// <param name="SessionKey">Key to be protected by the lock X. It will be recovered by any of the
clients in the supplied list.</param>
        /// <param name="RequestingClients">Clients requesting the current content. Only these clients will
be able to recover the session key, based on the lock X.</param>
        /// <returns>The lock X (see the Chinese Remainder Theorem)</returns>
        public BigInteger ComputeLockX(string SessionKey, List<string> RequestingClients)
        {
            RSACryptoServiceProvider rsa = new RSACryptoServiceProvider();
            ASCIIEncoding ae = new ASCIIEncoding();
            SortedList<string, BigInteger> Rs = new SortedList<string, BigInteger>();

            // Compute the R coeficient for each client, like so: Ri = SessionKey encrypted with the public
key of the client i
            foreach (string rc in RequestingClients)
            {
                try
                {
                    ContentClient client = RegisteredContentClients[rc];
                    // Get the public key for the current client
                    RSAParameters ClientPublicKey = client.PublicKey;
```

```
                    // Initialize our local RSA provider so we can work with it
                    rsa.ImportParameters(ClientPublicKey);
                    // Encrypt the session key
                    byte[] R = rsa.Encrypt(ae.GetBytes(SessionKey), false);
                    // Store R in a list of our own
                    Rs[client.Name] = new BigInteger(R);
                }
                catch (KeyNotFoundException)
                {
                    throw new ArgumentException("Client " + rc + " is not registered with the Content
Master.");
                }
            }
            // We have the R values, the N values (generated at client registration), so we can finally
compute the lock X using the Chinese Remainder Theorem
            ChineseRemainderTheorem crt = new ChineseRemainderTheorem();
            foreach (string client in RequestingClients)
            {
                crt.AddLinearCongruence(Rs[client], RegisteredContentClients[client].N);
            }
            return crt.Solve();
        }

        /// <summary>
        /// Returns the current ContentID then increments it.
        /// </summary>
        protected int GetNewContentID()
        {
            return ContentID++;
        }

        /// <summary>
        /// Gets the public key of the content master
        /// </summary>
        /// <returns>The public key</returns>
        public RSAParameters PublicKey
        {
            get
            {
                return ContentMasterRSA.ExportParameters(false);
            }
        }
        #endregion


    }
}
```

5.2.3 Operaţii matematice

Operaţiile matematice care implică folosirea numerelor foarte mari precum şi clasa răspunzătoare pentru rezolvarea congruenţelor liniare sunt grupate într-un modul specializat. Pentru lucrul cu numere foarte mari s-a folosit proiectul BigInteger [99], care reprezintă numerele sub formă de string. Rezolvarea congruenţelor liniare se face cu clasa LinearCongruence, redată mai jos.

```
namespace SCADDICMath
{

    /// <summary>
    /// The LinearCongruence class solves a linear congruence, as follows:
    ///
    /// If A and B are any integers and N is a positive integer, the linear congruence is:
    /// Ax ≡ B (mod N)
    ///
    /// Conditions:
    /// B is divided by GCD(A,N) where GCD stands for Greatest Common Divider.
    ///
    /// Solutions:
    /// The set of all solutions is {x0 + k * N/d} where k is an integer number and d = GCD(A,N).
    /// Keep only the positive solution less than N.
    ///
    /// x0 is calculated as follows:
```

```
    /// Using the Extended Euclidean Algorithm, find tow integers R and S such that R * A + S * N = d.
    /// x0 is then R * B / d
    ///
    /// The total number of valid solutions is d.
    /// </summary>
    /// <seealso
cref="http://en.wikipedia.org/wiki/Linear_congruence_theorem#Solving_a_linear_congruence"/>
    public class LinearCongruence
    {
        #region Congruence Parameters
        /// <summary>
        /// A
        /// </summary>
        public BigInteger A
        {
            get;
            set;
        }

        /// <summary>
        /// B
        /// </summary>
        public BigInteger B
        {
            get;
            set;
        }

        /// <summary>
        /// N
        /// </summary>
        public BigInteger N
        {
            get;
            set;
        }
        #endregion

        #region Public Constructors
        public LinearCongruence(BigInteger B, BigInteger N)
        {
            this.A = 1;
            this.B = B;
            this.N = N;
        }

        public LinearCongruence(BigInteger A, BigInteger B, BigInteger N)
        {
            this.A = A;
            this.B = B;
            this.N = N;
        }
        #endregion

        /// <summary>
        /// Solves a linear congruence in the form Ax ≡ B (mod N)
        /// </summary>
        /// <returns>An array containing the solutions (if any).</returns>
        public BigInteger[] Solve()
        {
            BigInteger d = A.gcd(N);
            // Check whether we have solutions
            if(B %d != 0) throw new ArgumentException("There are no solutions for linear congruence " + A +
"x ≡ " + B + " (mod " + N + ").");
            // Apply the extended Euclidean algorithm to find out R and S
            BigInteger R, S;
            ExtendedGCD(A, N, out R, out S);
            // Calculate x0
            BigInteger x0 = R * (BigInteger)(B / d);
            // Generate the set of all solutions (there are exactly "d" solutions)
            BigInteger[] Solutions = new BigInteger[d.IntValue()];
            int iSol = 0, k = 0;        // This shows the next available index in the solution array

            BigInteger incr = (BigInteger)(N / d);
            while (iSol < d)
            {
                // Calculate the possible solution
                BigInteger Sol = x0 + (k++) * incr;
                // The solution must be positive and less than N (since we're talking modulo N)
                if ((Sol > 0) && (Sol < N))
                {
                    Solutions[iSol++] = Sol;
```

```
                }
            }
            // Return the solutions of the linear congruence
            return Solutions;
        }

        /// <summary>
        /// Implements the extended Euclidean algorithm.
        /// Finds r and s such that Ar + Bs = GCD(A,B) where GCD stands for the Greatest Common Divider.
        /// </summary>
        /// <param name="A">A</param>
        /// <param name="B">B</param>
        /// <param name="r">r</param>
        /// <param name="s">s</param>
        protected void ExtendedGCD(BigInteger A, BigInteger B, out BigInteger r, out BigInteger s)
        {
            BigInteger x = 0, y = 1, lastx = 1, lasty = 0;

            while (B != 0)
            {
                BigInteger temp = B;
                BigInteger quotient = (BigInteger)(A / B);
                B = A % B;
                A = temp;

                temp = x;
                x = lastx - quotient * x;
                lastx = temp;

                temp = y;
                y = lasty - quotient * y;
                lasty = temp;
            }
            r = lastx;
            s = lasty;
        }

        /// <summary>
        /// Implements the Euclidean algorithm (finds the greatest common divisor of two numbers).
        /// </summary>
        /// <param name="A">A.</param>
        /// <param name="B">B.</param>
        /// <returns>Greatest common divisor of A and B.</returns>
        protected BigInteger GCD(BigInteger A, BigInteger B)
        {
            BigInteger Temp;
            while (B != 0)
            {
                Temp = B;
                B = A % B;
                A = Temp;
            }
            return A;
        }

        #region Overriden Methods
        public override string ToString()
        {
            return A + "x ≡ " + B + " (mod " + N + ")";
        }
        #endregion
    }
}
```

Rezolvarea sistemelor de congruenţe liniare (teorema chineză a restului), se face cu clasa ChineseRemainderTheorem, redată mai jos.

```
namespace SCADDICMath
{
    /// <summary>
    /// The ChineseRemainderTheorem class solves a system of linear congruences such as:
    /// X ≡ A1 (mod m1)
    /// X ≡ A2 (mod m2)
    /// ...
    /// X ≡ Ar (mod mr)
    ///
    /// The solution X can be calculated as (a1 * b1 * M / m1 + ... + a2 * b2 * M / mr) mod M.
```

```
    /// where: M is defined as m1 * m2 * ... mr
    /// Each b term can be deduced by solving this linear congruence: b * M / m ≡ 1 (mod m)
    /// </summary>
    /// <seealso cref="http://mathworld.wolfram.com/ChineseRemainderTheorem.html"/>
    public class ChineseRemainderTheorem
    {
        #region Local variables
        protected List<LinearCongruence> LinCong;
        #endregion

        #region Public constructor
        public ChineseRemainderTheorem()
        {
            LinCong = new List<LinearCongruence>();
        }
        #endregion

        /// <summary>
        /// Adds a new linear congruence to the equation system, in the form x ≡ A (mod m)
        /// </summary>
        /// <param name="A">A</param>
        /// <param name="m">m</param>
        public void AddLinearCongruence(BigInteger A, BigInteger m)
        {
            LinCong.Add(new LinearCongruence(A, m));
        }

        public BigInteger Solve()
        {
            // All linear congruences must have been previously loaded
            if(LinCong.Count == 0) throw new ArgumentException("Please specify at least one linear
congruence.");
            // Calculate M = m1 * m2 *...* mr where r is the number of linear congruences we have in the
system
            BigInteger M = GetM();
            // Calculate Term = a1 * b1 * M / m1 + ... + ar * br * M / mr where r is the number of linear
congruences we have in the system
            BigInteger Term = 0;
            for (int r = 0; r < LinCong.Count; r++)
            {
                BigInteger a = LinCong[r].B;
                BigInteger m = LinCong[r].N;

                // We need to calculate the product for the current r: a * b * M / m
                // b is deduced by solving the linear congruence b * M / m ≡ 1 (mod m)
                LinearCongruence templc = new LinearCongruence((BigInteger)(M / m), 1, m);
                BigInteger[] solutions = templc.Solve();
                if (solutions.Length != 1) throw new ArgumentException("Equation is not solvable. Please
check whether all N's are pairwise coprime numbers.");
                // We should have just one solution
                BigInteger b = solutions[0];

                BigInteger Product = a * b * M / m;
                Term += Product;
            }
            // Now we can determine the solution of the equation system: X = Term mod M
            BigInteger X = Term % M;
            return X;
        }

        /// <summary>
        /// M is m1 * m2 * ... * mr where r is the number of equations
        /// </summary>
        protected BigInteger GetM()
        {
            BigInteger result = 1;
            foreach (LinearCongruence lc in LinCong)
            {
                result *= lc.N;
            }
            return result;

        }

    }
}
```

5.3 Măsurători de performanţă

Pentru a valida modelul teoretic şi pentru a demonstra fezabilitatea soluţiei propuse, prototipul a fost supus unor măsurători de performanţă. Performanţa a fost măsurată pentru faza de iniţializare (care are loc o singură dată, la pornirea sistemului) şi pentru faza de operare. Pe de o parte, faza de iniţializare are o importanţă secundară datorită caracterului său tranzitoriu, însă măsurarea acestor timpi ajută la formarea unei opinii legate de timpul de cold-start al unui dispozitiv portabil care ar implementa sistemul SCADDIC. Pe de altă parte, măsurarea performanţei în faza de operare oferă indicii precise asupra comportamentului sistemului în timpul funcţionării de fapt.

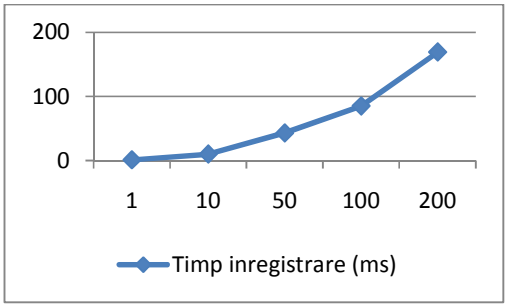
Codul C# al prototipului a fost compilat cu Visual Studio 2008 pe .NET Framework 3.5 şi a fost rulat pe un calculator portabil de tip Lenovo T60 (1,80 Ghz, 2 GB RAM). Timpul măsurat în diverse stadii reprezintă duratele necesare diverselor operaţii criptografice şi nu include timpul petrecut de conţinut în tranzit prin diverse medii de transmisie.

5.3.1 Faza de iniţializare

Faza de iniţializare a sistemului SCADDIC începe cu instanţierea Content Master-ului, operaţie urmată la scurt timp de **înregistrarea furnizorilor de conţinut** (Content Provider). Prin operaţia de înregistrare, furnizorii îşi anunţă prezenţa, participând ulterior la schema de împărţire a secretului şi la votul acordării de permisiuni clienţilor care solicită primirea unui conţinut digital.

Se observă că timpul necesar pentru înregistrarea furnizorilor de conţinut creşte cvasi-liniar cu numărul furnizorilor de conţinut,timpul mediu de înregistrare fiind de 0,9 ms / furnizor.

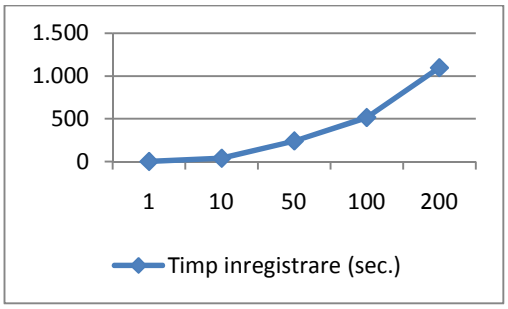
Numărul furnizorilor de conţinut	Timp de înregistrare (ms)
1	1
10	10
50	43
100	85
200	169



Tabel 3. Evoluţia timpului de înregistrare al furnizorilor de conţinut

Iniţializarea clienţilor presupune o serie de operaţii criptografice consumatoare de timp. Aceste operaţii se efectuează în mod normal la momentul fabricaţiei (în cazul dispozitivelor embedded), deoarece perechea de chei publică şi privată nu se modifică pe durata vieţii dispozitivului. În cazul prototipului SCADDIC acest lucru are loc la fiecare iniţializare a sistemului.

Număr de clienţi	Timp de înregistrare (s)
1	1.2
10	38.6
50	240.3
100	515.1
200	1097.1

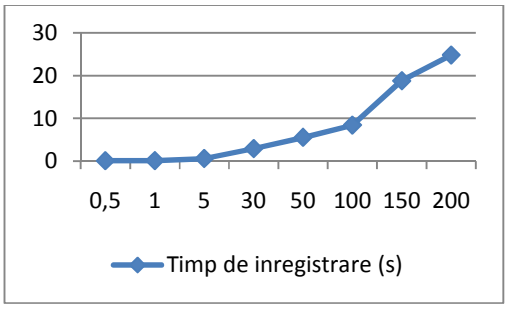


Tabel 4. Evoluţia timpului de înregistrare pentru clienţi

Timpul necesar pentru înregistrarea clienţilor este semnificativ mai lung decât cel necesar pentru înregistrarea furnizorilor de conţinut datorită operaţiilor cu criptografie asimetrică implicate. Timpul mediu de înregistrare pentru un client este de 4,1 sec.

Înregistrarea conţinutului este operaţia prin care un furnizor îşi anunţă intenţia de a face disponibil conţinutul digital aflat în posesia sa. Acest pas are loc după înregistrarea furnizorilor de conţinut, deoarece drepturile de distribuţie se pot partaja între furnizori. Pentru a evita scurgerile de informaţii şi pentru a proteja secretul conţinutului aflat în conservare, adică înainte de a fi oferit clienţilor, Content Master-ul va cripta fiecare conţinut cu o cheie aleatoare care ulterior va fi împărţită între furnizorii înregistraţi.

Dimensiune conţinut (MB)	Timp înregistrare (sec)
0.5	67
1	98
5	580
30	2918
50	5525
100	8393
150	18775
200	24794



Tabel 5. Timpul de înregistrare a conţinutului

Timpul necesar pentru înregistrarea conţinutului creşte cvasi-liniar cu dimensiunea. Timpul mediu de înregistrare este de 110 ms per MB.

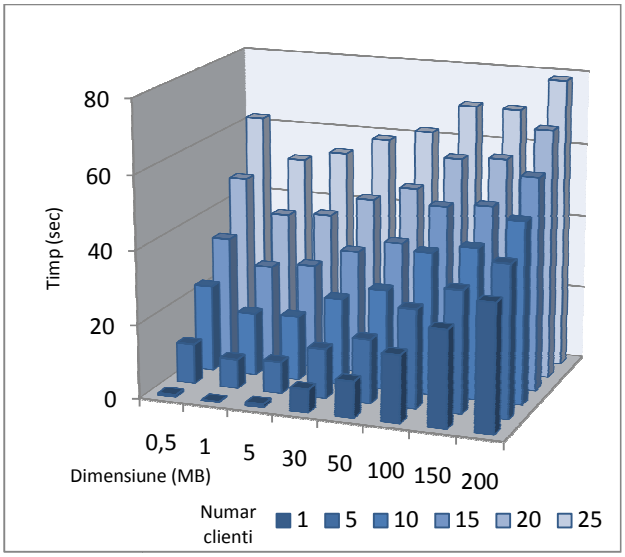
5.3.2 Faza de operare

Faza de operare presupune distribuţia conţinutului în sine, adică trimiterea conţinutului digital la clienţi. Distribuţia este o operaţie complexă care cuprinde următoarele task-uri:

1. Cererea autorizaţiei furnizorului de conţinut
2. Generarea cheii de sesiune
3. Calcularea lacătului X
4. Calcularea valorii de dispersie a conţinutului.
5. Criptarea conţinutului cu cheia de sesiune.

Evidenţierea performanţei pentru fiecare dintre aceste task-uri ar dilua măsurarea performanţei, de aceea distribuţia de conţinut a fost simulată ca un tot unitar într-o varietate de condiţii, începând de la cazul simplu de 1 client cu conţinut de distribuit de 0,5 MB până la 25 de clienţi cu conţinut de 200MB. Tabelul următor prezintă rezultatele măsurătorilor pe calculatorul de test, în diverse scenarii.

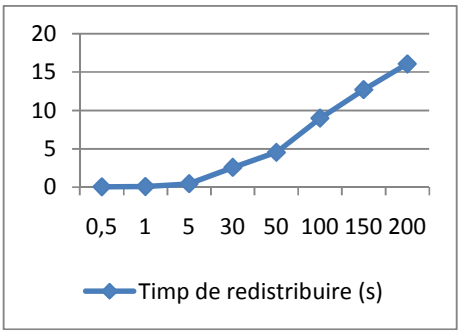
Număr clienţi / Dimensiune conţinut (MB)	1	5	10	15	20	25
0.5	0.68	10.86	23.48	33.52	47.58	62.11
1	0.21	7.96	17.12	26.85	38.41	51.23
5	0.97	8.63	17.54	28.34	39.38	54.00
30	6.37	13.47	23.6	33.37	44.80	58.77
50	9.90	17.39	27.27	36.84	48.84	61.86
100	1.31	26.81	38.55	47.91	58.06	69.89
150	2.28	33.14	41.00	49.07	58.95	69.79
200	34.56	41.24	49.19	57.85	67.71	78.62



Tabel 6. Timpul de distribuţie în funcţie de numărul clienţilor şi dimensiunea conţinutului digital

Timpul necesar pentru pregătirea conţinutului pentru difuzare (broadcast) creşte liniar cu numărul de clienţi şi subliniar cu dimensiunea datelor de distribuit. Odată cu creşterea numărului de clienţi, sistemul de distribuţie este avantajat de dimensiuni mari ale conţinutului.

Dimensiune conţinut (MB)	Timp de redistribuţie (s)
0.5	0.05
1	0.09
5	0.45
30	2.57
50	4.53
100	8.98
150	12.69
200	16.04



Tabel 7. Timpul de redistribuţie a conţinutului

Redistribuirea conţinutului este o operaţie care implică două părţi, emiţătorul şi receptorul. Emiţătorul este entitatea în posesia căreia se află conţinutul şi în plus are drept de redistribuire în mod legal. Receptorul reprezintă entitatea care primeşte conţinutul în baza unei înţelegeri prealabile cu emiţătorul. În cazul redistribuirii conţinutului nu este nevoie să se calculeze lacătul X, însă pentru protejarea conţinutului acesta se va cripta cu o cheie de sesiune care la rândul ei este criptată cu cheia publică a receptorului.

Timpul necesar pentru redistribuirea conţinutului creşte liniar cu dimensiunea acestuia. Timpul mediu de redistribuţie este de 88 ms per MB.

5.4 Avantaje ale sistemului de distribuţie propus

Pentru a evalua performanţa sistemului de distribuţie propus şi avantajele sale din punctul de vedere al scalabilităţii şi pentru a putea raporta rezultatele la implementarea din [63], este necesară definirea unei metrici după care să se evalueze performanţa. Pentru scopul paragrafului de faţă, convenim să numim metoda implementată în [63] „unicast” datorită faptului că transmisia datelor se

face în mod punct la punct, iar implementarea propusă în această lucrare se va numi „broadcast" datorită faptului că un număr multiplu de clienţi poate fi deservit la un moment dat.

Metrica pe care o propunem aici ia în calcul o serie de parametri comuni celor două implementări (unicast şi broadcast), după cum urmează:

1. Timpul de generare a cheii de sesiune
2. Timpul necesar criptării conţinutului cu cheia de sesiune
3. Timpul necesar criptării cheii de sesiune (fapt care înseamnă crearea lacătului X care va fi ulterior folosit pentru recuperarea cheii de sesiune)
4. Timpul necesar distribuţiei conţinutului criptat şi a cheii de sesiune, de asemenea criptată

Astfel, timpul total de distribuţie a unui conţinut digital este exprimat de următoarea formulă:

$$T_{dist} = \sum_{i=1}^{N} Tsk_i + \sum_{i=1}^{N} Tce_i + \sum_{i=1}^{M} Tx_i + \sum_{i=1}^{N} Tsend_i$$

(Ecuația 11)

Unde:

- T_{dist} – Timpul total necesar pentru a transmite toate instanţele conţinutului
- T_{sk} – Timpul necesar pentru a genera cheia de sesiune
- T_{ce} – Timpul necesar pentru a cripta conţinutul cu cheia de sesiune
- T_x – Timpul necesar pentru a general lacătul C sau pentru a cripta cheia de sesiune cu cheia publică a receptorului (în cazul redistribuţiei)
- T_{send} – Timpul necesar pentru a transmite o instanţă a conţinutului către client
- N – Numărul de instanţe ale conţinutului care trebuie transmise
- M – Numărul de clienţi la care conţinutul trebuie transmis

Pentru scenariul broadcast, numărul instanţelor de conţinut care trebuie transmise este N = 1, deci (Ecuația **11**) devine:

$$T_{broadcast} = Tsk + Tce + \sum_{i=1}^{M} Tx_i + Tsend$$

(Ecuația 12)

Unde:

- $T_{broadcast}$ – Timpul total necesar pentru a difuza conţinutul

Pentru scenariul unicast, numărul de instanţe ale conţinutului care trebuie transmise este egal cu numărul clienţilor, astfel încât N = M. (Ecuația **11**) devine astfel:

$$T_{unicast} = \sum_{i=1}^{N} (Tsk_i + Tce_i + Tx_i + Tsend_i)$$

(Ecuația 13)

Unde:

- $T_{unicast}$ – Timpul total necesar pentru a trimite conţinutul fiecărui client în parte (unicast)

Cele două ecuaţii scot la iveală faptul că scenariul broadcast poate fi avantajat din punct de vedere al timpului, în condiţii de încărcare identică. Pentru a avea o imagine asupra diferenţelor între cele două tipuri de sisteme de distribuţie, simularea s-a făcut variind dimensiunea conţinutului şi numărul de clienţi care solicită acelaşi conţinut la un moment dat. Ca urmare a simulărilor premergătoare cu module din cadrul sistemului SCADDIC, s-au obţinut următoarele:

- Cheia de sesiune se generează într-un timp de 2 ms.
- Conţinutul este criptat la o rată de 73 ms per MB, folosind un algoritm asimetric.
- Lacătul X se calculează în 117 ms pentru un client şi are o creştere liniară cu numărul de clienţi.
- Pentru unicast, lacătul X nu este calculat şi este înlocuit de o criptare cu chei asimetrice, care se consideră că durează tot 117 ms.
- Viteza legăturii dintre emitent şi receptor este de 200KB/sec, o valoare considerată tipică pentru un dispozitiv embedded de astăzi.

Timpii redaţi mai jos au fost obţinuţi după medierea rulărilor repetate ale simulării.

	0.5MB	1MB	5MB	30MB	50MB	100MB	150MB	200MB
Broadcast (1 client)	3	5	26	156	260	519	779	1039
Unicast (1 client)	3	5	26	156	260	519	779	1039
Broadcast (5 clienţi)	6	8	29	159	263	522	782	1042
Unicast (5 clienţi)	14	27	131	780	1299	2597	3895	5194

Broadcast (10 clienţi)	14	17	38	168	271	531	791	1050
Unicast (10 clienţi)	27	53	261	1559	2598	5194	7791	10387
Broadcast (50 clienţi)	295	298	318	448	552	812	1071	1331
Unicast (50 clienţi)	137	267	1305	7796	12989	25972	38954	51937
Broadcast (100 clienţi)	1173	1175	1196	1326	1430	1689	1949	2209
Unicast (100 clienţi)	273	533	2610	15593	25979	51944	77909	103874

Tabel 8. Performanţa în secunde pentru diverse tipuri de conţinut şi număr de clienţi

Pentru un singur client, comportamentul în modul broadcast este acelaşi cu cel din modul unicast, deci nu se observă un câştig de viteză (Fig. 37). Modul broadcast în sine presupune existenţa a cel puţin doi clienţi care solicită simultan exact acelaşi conţinut, deci în cazul unei singure cereri modul broadcast se reduce ca funcţionare la modul unicast.

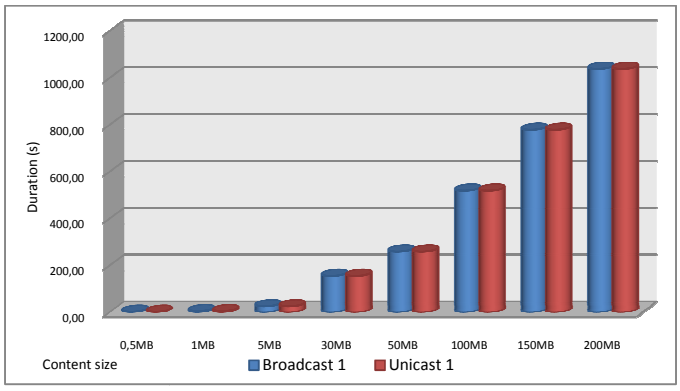


Fig. 37 Comparaţie Broadcast – Unicast pentru un client

Creşterea numărului de clienţi care solicită simultan acelaşi conţinut digital duce la creşterea performanţei sistemului broadcast în comparaţie cu cel unicast, care în acest caz efectuează o singură dată operaţiile criptografice costisitoare. Din figurile Fig. 38 - Fig. 41 se observă creşterea tot mai accentuată a performanţei sistemului broadcast odată cu creşterea numărului de clienţi deserviţi.

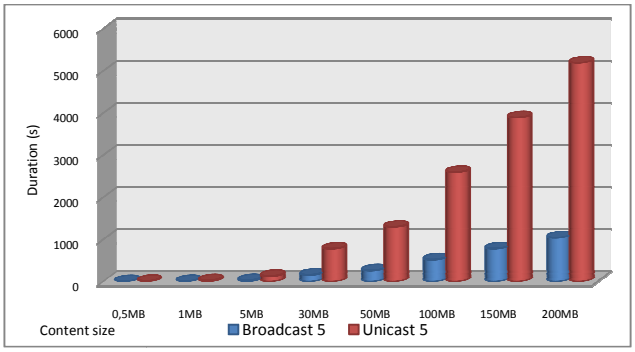


Fig. 38 Comparaţie Broadcast – Unicast pentru 5 clienţi

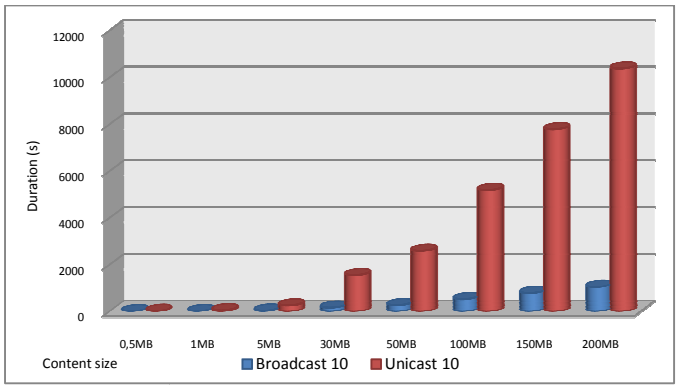


Fig. 39 Comparaţie Broadcast – Unicast pentru 10 clienţi

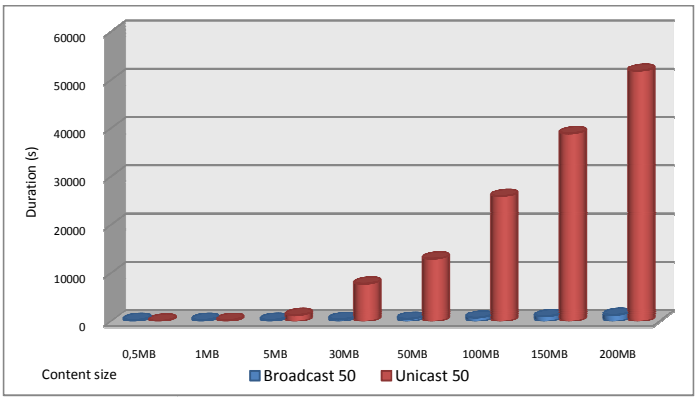


Fig. 40 Comparaţie Broadcast – Unicast pentru 50 de clienţi

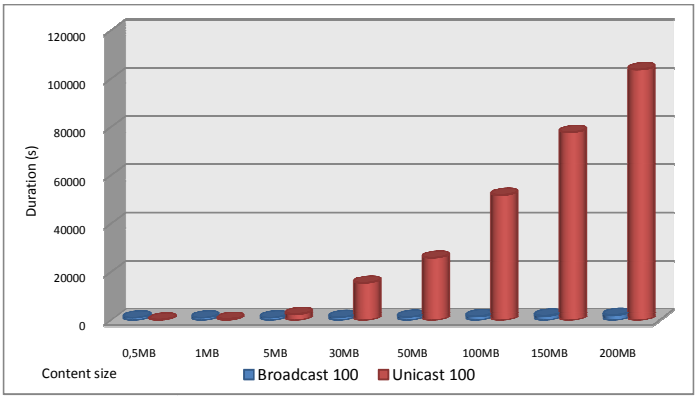


Fig. 41 Comparaţie Broadcast – Unicast pentru 100 de clienţi

5.5 Concluzii

Sistemele de distribuţie digitală a conţinutului existente astăzi sunt în general orientate pe deservirea unui singur client la un moment dat. Arhitecturile de acest tip prezintă neajunsul de a nu fi scalabile, deoarece operaţiile de criptare şi transmitere a conţinutului digital se repetă pentru fiecare client în parte.

Arhitectura propusă în cadrul acestui capitol se construieşte pe sistemul de distribuţie prezentat în [63] şi are avantajul unei performanţe ameliorate şi a posibilităţii de arbitrare criptografică a drepturilor partajate de furnizori. Performanţa arhitecturii este superioară datorită modelului de distribuţie de tip broadcast. Acesta presupune codificarea conţinutului prin procedee criptografice, permiţând astfel ca transmiterea sa să fie simultană şi securizată către mai mulţi clienţi. În plus, acest model de distribuţie permite activarea rolului de proxy, adică clienţi aflaţi mai aproape de centrul de distribuţie pot acţiona ca releu de distribuţie, degrevând serverul de sarcini.

Avantajele sistemului de distribuţie digitală a conţinutului propus pot fi sumarizate după cum urmează:

- **Scalabilitate** – În momentele de vârf, când serverul este asaltat de cereri performanţa sistemului de broadcast propus este foarte bună deoarece distribuţia de conţinut se face simultan la un număr de clienţi cu cereri similare, acumulate la intervale de timp prestabilite. Sistemul propus poate fi de până la câteva ordine de mărime mai scalabil decât varianta din [63], funcţie de factori cum sunt numărul de clienţi, dimensiunea conţinutului de distribuit şi viteza conexiunii dintre client şi server. Acest fapt a fost dovedit experimental.

- **Distribuţie asincronă** – Deoarece conţinutul digital este criptat şi este disponibil doar unor clienţi selectaţi de către server pe diverse criterii, fluxul digital poate fi răspândit (broadcast) prin reţea fără a exista pericolul ca un client neautorizat să-l poată reda. Sarcina de distribuţie care revine în mod tradiţional serverului poate fi preluată cu succes de către orice clienţi aflaţi mai aproape de receptor. În acest fel, serverul este liber să efectueze alte operaţii rezultând un factor de scalabilitate bun.
- **Colaborare între deţinătorii de copyright** – Uneori, entităţi diferite pot deţine drepturile asupra diverselor părţi ale conţinutului digital (coloană sonoră, imagine, traducere, prezentare grafică, etc.), accesul la conţinut făcându-se când toate entităţile îşi dau acordul. Alteori, anumite entităţi pot cere a fi informate asupra modului în care conţinutul digital este accesat. În toate aceste cazuri arhitectura propusă este capabilă să partajeze cu succes drepturile asupra conţinutului prin tehnica „secret spliting" şi să ofere acces fizic la acesta doar când participanţii sunt cu toţii de acord.

Tot în cadrul acestui capitol s-a propus o metrică pentru evaluarea performanţei sistemului de distribuţie. Aceasta ţine cont de parametri criptografici din cadrul sistemului şi care se adaptează funcţie de modelul analizat, mai precis unicast sau broadcast. Această metrică reprezintă numitorul comun ce era necesar pentru a putea compara două sisteme competitive, similare ca funcţionalitate dar diferite ca design.

6. Concluzii finale și perspective

6.1 Concluzii finale

În societatea informatizată a zilelor noastre, securitatea şi buna funcţionare a sistemelor de comunicaţie este o problemă de primă importanţă. Afacerile noastre, confortul nostru şi uneori chiar şi viaţa depind în mare măsură de comunicaţie, de aceea o atenţie deosebită trebuie acordată disponibilităţii şi scalabilităţii sistemelor care asigură comunicaţia.

Atacurile reprezintă acţiuni subversive menite să compromită buna funcţionare a sistemelor, fiind lansate de indivizi sau grupări care urmăresc obţinerea de avantaje materiale sau de alt fel, în detrimentul masei de utilizatori. Deşi la nivel legal există metode de constrângere şi de pedepsire a acestor practici, acest lucru nu este suficient în majoritatea cazurilor, deoarece atacul poate avea loc de la distanţă, în zone cu jurisdicţie diferită, iniţiatorii putându-şi pierde astfel urma. Din punct de vedere tehnic sistemele trebuie să fie pregătite pentru a reduce cât mai mult suprafaţa de atac şi de a limita efectele unui atac care a avut deja loc. Securizarea căilor de comunicaţie este un domeniu în care contribuţiile din literatură sunt variate. Unul dintre domeniile în care încă există carenţe în abordare este creşterea disponibilităţii şi asigurarea scalabilităţii sistemelor de comunicaţie, această nişă fiind acoperită de prezenta lucrare de cercetare.

Lucrarea de faţă a avut în vedere un orizont larg de sisteme:

- Sistemele de autentificare
- Sistemele de tip Single Sign-On (SSO)
- Reţelele de comunicaţie mobilă de tip GSM
- Sistemele de distribuţie a conţinutului digital

În cadrul lucrării s-a făcut o descriere a fiecărui sistem amintit şi a scos în evidenţă deficienţele existente în prezent, astfel:

- Sistemele de autentificare, sistemele Single Sign-On (SSO) şi reţelele GSM sunt vulnerabile la atacuri de tip DoS. Vulnerabilitatea apare din cauza lipsei de control asupra resurselor alocate participanţilor la comunicaţie şi are de cele mai multe ori ca rezultat degradarea severă a serviciului sau blocarea lui pe perioada desfăşurării atacului.
- Sistemele de distribuţie a conţinutului actuale sunt puţin scalabile şi indirect sunt susceptibile la atacuri DoS.

Contribuţiile aduse de această lucrare cu privire la detectarea timpurie şi împiedicarea atacurilor de tip DoS pot fi sumarizate după cum urmează:

- Au fost dezvoltate tehnologiile Threshold Puzzles şi Adaptive Threshold Puzzles, pentru a acoperi situaţiile în care tehnologia Client Puzzles nu este eficientă [11] [12] [13].
- S-au introdus două versiuni ale schimbului de mesaje Handshake din cadrul protocolului SSL, astfel încât acesta să suporte tehnologiile Threshold Puzzles şi Adaptive Threshold Puzzles.
- S-a evidenţiat posibilitatea ca atacurile DoS să fie detectate cu o anumită probabilitate înainte ca acestea să aibă loc prin aplicarea inferenţei bayesiene [13].
- S-a introdus SSO-Sense, un modul de estimare euristică a nivelului de risc pentru sistemele de autentificare [13].

În ceea ce priveşte siguranţa şi disponibilitatea reţelelor GSM, lucrarea aduce următoarele contribuţii:

- Vulnerabilităţile sistemului GSM au fost evaluate cu o metodă de clasificare numită DREAD. Rezultatul deloc surprinzător a fost că cea mai periculoasă ameninţare la adresa GSM este atacul DoS (denial of service) [16].
- S-a arătat exact cauza pentru care sistemul GSM este vulnerabil la atacuri DoS, şi anume lipsa preautentificării dispozitivelor mobile care pot cere alocarea de resurse în mod repetat [103].
- S-a arătat de asemenea că un singur atacator este capabil de a dezactiva o întreagă celulă GSM şi în anumite cazuri chiar şi celulele adiacente [103].
- Deoarece nu sunt implicate costuri financiare (nu se efectuează nici un apel în sine), costul efectiv al lansării un atac devastator este zero din punctul de vedere al plăţilor către operatorul vizat [103] [14].
- În scopul creşterii rezistenţei la atacuri DoS, s-a propus introducerea preautentificării terminalelor GSM în cazul iniţierii unui apel de către acestea. Preautentificarea impune introducerea în protocolul Channel Assignment a unui mesaj nou de tip baliză numit PREAUTHENTICATION BEACON care transmite terminalelor mobile spre rezolvare o mică problemă, după modelul Threshold Puzzles [14] [16].

Lucrarea a introdus de asemenea şi o arhitectură de distribuţie a conţinutului digital cu următoarele avantaje [15]:

- **Scalabilitate** – În momentele de vârf, când serverul este asaltat de cereri performanţa sistemului de broadcast propus este foarte bună

deoarece distribuţia de conţinut se face simultan la un număr de clienţi cu cereri similare, acumulate la intervale de timp prestabilite.

- **Distribuţie asincronă** – Deoarece conţinutul digital este criptat şi este disponibil doar unor clienţi selectaţi de către server pe diverse criterii, fluxul digital poate fi răspândit (broadcast) prin reţea fără a exista pericolul ca un client neautorizat să-l poată reda. Sarcina de distribuţie care revine în mod tradiţional serverului poate fi preluată cu succes de către orice clienţi aflaţi mai aproape de receptor. În acest fel, serverul este liber să efectueze alte operaţii rezultând un factor de scalabilitate bun.
- **Colaborare între deținătorii de copyright** – Uneori, entităţi diferite pot deţine drepturile asupra diverselor părţi ale conţinutului digital (coloană sonoră, imagine, traducere, prezentare grafică, etc.), accesul la conţinut făcându-se când toate entităţile îşi dau acordul. Alteori, anumite entităţi pot cere a fi informate asupra modului în care conţinutul digital este accesat. În toate aceste cazuri arhitectura propusă este capabilă să partajeze cu succes drepturile asupra conţinutului prin tehnica „secret spliting" şi să ofere acces fizic la acesta doar când participanţii sunt cu toţii de acord.

De asemenea, s-a propus o metrică pentru evaluarea performanţei sistemului de distribuţie. Aceasta ţine cont de parametri criptografici din cadrul sistemului şi care se adaptează funcţie de modelul analizat, mai precis unicast sau broadcast. Această metrică reprezintă numitorul comun necesar pentru a putea compara sistemele similare ca funcţionalitate dar diferite ca design [17].

6.2 Perspective

Dacă în anii 1990 şi 2000 spectrul radio a fost dominat de generaţiile 2/2.5 (GPRS-EDGE) şi 3/3.5 (3G-HSDPA) de transmisii de date şi comunicaţii mobile, în perioada următoare spectrul radio va face loc noului venit LTE. Această nouă tehnologie permite rate de download de până la 100 Mbit/sec, uplink de 50 Mbit/sec şi timp dus-întors pe interfaţa radio de sub 10 ms. De asemenea, tehnologia LTE suportă interoperabilitatea cu infrastucturile GSM, CDMA şi WCDMA existente.

Cercetarea din perioada următoare vizează următoarele aspecte ale securităţii LTE:

- Analiza autentificării şi distribuţiei cheilor
- Protecţia canalelor de semnalizare în Core Network (NAS) şi Radio Network (RRC) şi analiza vulnerabilităţii la atacuri DoS
- Scalabilitatea şi capacitatea de ameliorare în caz de atac DoS
- Protecţia planului utilizatorului

Abrevieri şi acronime

- **ACL** Access Control List
- **AES** Application Environment Specification
- **ANSI** American National Standards Institute
- **APA** Authentication and Privilege Attribute
- **API** Application Programming Interface
- **AS** Authentication Server
- **ASN.1** Abstract Syntax Notation 1
- **ATM** Asynchronous Transfer Mode

- **BAN** Burrows, Abadi, Needham
- **Bellcore** Bell Communications Research
- **BER** Basic Encoding Rules
- **BFI** Swiss Federal Office of Information Technology and Systems
- **BTS** Base Transceiver Station (folosit în terminologia GSM)

- **CA** Certification Authority
- **CAA** Certification Authority Agent
- **CAE** Common Applications Environment
- **CAT** Common Authentication Technology
- **CBC** Cipher Block Chaining
- **CCITT** Consultative Committee on International Telegraphy and Telephony (now ITU-T)
- **CD** Compact Disc
- **CDC** Certificate Distribution Center
- **CDMF** Commercial Data Masking Facility
- **CDS** Cell Discovery Service
- **CEC** Commission of the European Communities
- **CFB** Cipher Feedback
- **CSF** Cryptographic Support Facility
- **CV** Control Value

- **DAC** Discretionary Access Control
- **DASS** Distributed Authentication Security Service
- **DCE** Distributed Computing Environment
- **DEC** Digital Equipment Corporation
- **DES** Data Encryption Standard
- **DIT** Directory Information Tree

- **DNS** Domain Name Service
- **DoC** U.S. Department of Commerce
- **DoD** U.S. Department of Defense
- **DoD** U.S. Department of State
- **DOS** Atac de tip Denial of Service
- **DSA** Digital Signature Algorithm
- **DSS** Digital Signature Standard
- **DSS** Domain Security Standard
- **DSSA** Distributed System Security Architecture
- **DTI** Directory Information Tree

- **ECB** Electronic Code Book
- **ECMA** European Computer Manufacturers Association
- **EDI** Electronic Data Interchange
- **EES** Exponential Electronic Signature
- **EISS** European Institute for System Security
- **EKE** Encrypted Key Exchange
- **EPAC** Extended Privilege Attribute Certificate
- **EU** European Union

- **FAQ** Frequently Asked Questions
- **FEAL** Fast Encryption Algorithm
- **FIPS** Federal Information Processing Standard
- **FTP** File Transfer Protocol

- **GDA** Global Domain Agent
- **GDS** Global Domain Service
- **GNY** Gong, Needham, Yahalom
- **GSM** Global System for Mobile Communications
- **GPRS** General Packet Radio Service
- **GSS-API** Generic Security Service API

- **HP** Hewlett-Packard

- **IAM** Institute for Computer Science and Applied Mathematics
- **IBM** International Business Machines Corporation
- **ICL** International Computers Limited
- **ICSI** International Computers Science Institute
- **IDEA** International Data Encryption Algorithm
- **IDS** Inter Domain Service
- **IEC** International Electrotechnical Committee
- **IEEE** Institute of Electrical and Electronic Engineers
- **IETF** Internet Engineering Task Force
- **IP** Internet Protocol

- **IPSEC** IP Security Protocol
- **IPST** IP Secure Tunnel Protocol
- **IS** International Standard
- **ISO** International Organization for Standardization
- **ISODE** ISO Development Environment
- **IT** Information Technology
- **ITU-T** International Telecommunication Union

- **JTC1** Joint Technical Committee 1

- **KDC** Key Distribution Center
- **KDS** Key Distribution Server
- **KEK** Key Encryption Key
- **KTC** Key Translation Center

- **LAN** Local Area Network
- **LEAF** Login Enrollment Agent Facility
- **LLC** Logical Link Control
- **LRA** Local Registration Authority
- **LTE** Long Term Evolution (WiMAX succesor)

- **MAC** Message Authentication Code
- **MAN** Metropolitan Area Network
- **MD** Message Digest
- **MDC** Modification Detection Code
- **MHS** Message Handling System
- **MIB** Management Information Base
- **MIC** Message Integrity Code
- **MIT** Massachusetts Institute of Technology
- **MKMP** Modular Key Management Protocol

- **NBS** National Bureau of Standards
- **NCSC** National Computer Security Center
- **NCSL** National Computer Systems Laboratory
- **NetSP** Network Security Program
- **NII** National Information Infrastructure
- **NIST** National Institute of Standards and Technology
- **NLSP** Network Layer Security Protocol
- **NSA** National Security Agency

- **OFB** Output Feedback
- **OMC** Operations Management Center
- **OSF** Open Software Foundation
- **OSI** Open Systems Interconnection

- **OSI-RM** OSI Reference Model
-
- **PAC** Privilege Attribute Certificate
- **PAS** Privilege Attribute Server
- **PC** Personal Computer
- **PEM** Privacy Enhanced Mail
- **PGP** Pretty Good Privacy
- **PIN** Personal Identification Number
- **PKCS** Public Key Cryptography Standard
- **PKM** Public Key Management
- **PKP** Public Key Partners
- **PPID** Primary Principal Identifier
- **PT** Privilege Ticket
- **PTGT** Privilege Ticket Granting Ticket
- **PV** Protection Value
- **PVF** PAC Validation Facility

- **RACF** Resource Access Control Facility
- **RFC** Request for Comments
- **RFT** Request for Technology
- **ROM** Read Only Memory
- **RPC** Remote Procedure Call
- **RSA** Rivest, Shamir, Adelman

- **SACM** Secure Association Context Manager
- **SELANE** Secure Local Area Network Environment
- **SESAME** Secure European System for Applications in a Multi-vendor Environment
- **SHA** Secure Hash Algorithm
- **SHS** Secure Hash Standard
- **SKIA** Secure Key Issuing Authority
- **SLC** Secured Logon Coordinator
- **SMIB** Security Management Information Base
- **SMS** Service Management System
- **SNG** Secured Network Gateway
- **SNI** Siemens Nixdorf Informationssysteme
- **SNMP** Simple Network Management Protocol
- **SNP** Secure Network Programming
- **SPKM** Simple Public-key GSS-API Mechanisms
- **SSE** Software and Systems Engineering (SSE) Ltd.
- **SSO** Single Sign-On

- **TA** Trusted Authority
- **TAN** Transaction Authentication Number
- **TCB** Trusted Computing Base
- **TCP** Transport Control Protocol

- **TCP SYN** Atac prin epuizarea resurselor asupra protocolului TCP
- **TESS** The Exponential Security System
- **TGS** Ticket Granting Server
- **TGT** Ticket Granting Ticket
- **TLSP** Transport Layer Security Protocol
- **TVP** Time-Variant Parameter

- **UID** User Identification
- **UUID** Universal Unique Identifier
- **URL** Uniform Resource Locator
- **US** User Sponsor
- **U.S.** United States

- **WiMAX** Worldwide Interoperability for Microwave Access
- **WG** Working Group
- **WWW** World Wide Web

Bibliografie

[1] 3rd Generation Partnership Project – *Specification of the GSM-MILENAGE Algorithms: An example algorithm set for Authentication and Key Generation functions A3 and A8*, http://www.gsmworld.com/using/algorithms/docs/ 55205-600.pdf

[2] Alcatel University – *Introduction to the Alcatel GSM Network*, 2003

[3] R. Anderson, M. Kuhn, "Tamper Resistance – A Cautionary Note", Proceedinggs of the 2nd Usenix Workshop on Electronic Commerce, pages 1-11, November 1996.

[4] Adam Beck, Ulf Moller, Anton Stiglic – *Traffic analysis attacks and trade-offs in anonymity providing systems,* Information Hiding, 4th International Workshop, IHW 2001, volume 2137 of Lecture Notes in Computer Science, pages 245-257, Springer-Verlag, Berlin 2001

[5] P. Barford, D. Plonka - *Characteristics of network traffic flow anomalies*, in Proc 1st ACM SIGCOMM Internet Meas. Workshop, New York, 2001, pp. 69–74.

[6] E. Barkan, E. Biham, N. Keller – *Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication,* Advances in Cryptology - CRYPTO 2003, Springer Berlin / Heidelberg, Pag. 600, ISBN 978-3-540-40674-7

[7] S. M. Bellovin, M. Merritt – *Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks*, Proceedings of the IEEE Symposium on Security and Privacy, IEEE Computer Society Press, Los Alamitos, CA, 1992

[8] S. M. Bellovin, M. Merritt – *Augumented Encrypted Key Exchange*, Proceedings of the 1st ACM Conference on Communications and Computing Security, November 1993

[9] Steven M. Bellovin, Michael Merritt – *Limitations of the Kerberos Authentication System,* AT&T Bell Labs

[10] Valer BOCAN - *Developments in DOS research and mitigating technologies*, Periodica Politehnica, Transactions on Automatic Control and Computer Science, Vol. 49 (63), 2004

[11] Valer BOCAN - Threshold Puzzles. The Evolution of DoS-Resistant Authentication, Periodica Politehnica, Transaction on Automatic Control and Computer Science Vol. 49 (63), 2004, ISSN 1224-600X

[12] Valer BOCAN, Mihai Fagadar-Cosma - Adaptive Threshold Puzzles, EUROCON - The International Conference on "Computer as a tool", Belgrade, Serbia and Montenegro, 2005

[13] Valer BOCAN, Mihai Făgădar-Cosma – Towards DoS-resistant Single Sign-On Systems, EUROCON, Belgrade, Serbia, 2005

[14] Valer BOCAN, Vladimir CREŢU – *Mitigating Denial of Service Threats in GSM Networks*, The International Conference on Availability, Reliability and Security (ARES), Wien, 2006

[15] Valer BOCAN, Mihai Făgădar-Cosma – *Scalable and Secure Architecture for Digital Content Distribution*, International Conference on Software, Telecommunications and Computer Networks, Croatia, 2006

[16] Valer BOCAN, Vladimir CREŢU – *Threats and Countermeasures in GSM Networks*, Journal of Networks, Academy Publishers, ISSN 1796-2056

[17] Valer BOCAN, Vladimir CREŢU - *SCADDIC: The Implementation and Performance of a Scalable and Secure Architecture for Digital Content Distribution*, 7th Conference on Communications, Military Technical Academy, Bucharest, 2008

[18] Alan Burnett – *Securing the Wireless Internet*, Roke Manor Research Ltd, UK, 2003

[19] M. Burrows, M. Abadi, R. Needham – *"A Logic of Authentication"*, ACM Operating Systems Review, Vol. 23, 1989

[20] Computer Emergency Response Team - *CERT advisory CA-2000.01 Denial of service developments*, 2000 (http://www.cert.org/advisories/CA-2000-01.html)

[21] G. A. Champine, D. E. Geer, W. N. Ruh – *"Project Athena as a Distributed Computer System",* IEEE Computer, Vol. 23, September 1990

[22] G. A. Champine – *"MIT Project Athena – A Model for Distributed Computing"*, Digital Press, 1991

[23] E. Charniak - *Bayesian Networks without Tears*, *AI Magazine*, Winter 1991, pp. 50–63.

[24] David L. Chaum – *Untraceable electronic mail, return addresses and digital pseudonims*, Communications of the ACM, 24(2):84-90, 1981

[25] G.-H. Chiou and W.-T. Chen – "Secure Broadcasting Using the Secure Lock", *IEEE Trans. Software Eng.*, vol. 15, no. 8, pp. 929-934, August 1989.

[26] Jan de Clercq – *Single Sign-On Architectures*, Infrastructure Security, International Conference, InfraSec 2002, Bristol, UK, 2002

[27] Scott A. Crosby, Dan S. Wallach – *Denial of Service via Algorithmic Complexity Attacks*, Proceedings of the12th USENIX Security Symposium, 2003

[28] Drew Dean, Adam Stubblefield – *Using Client Puzzles to Protect TSL*, http://www.csl.sri.com/users/ddean/papers/usenix01b.pdf, Proceedings of the10th USENIX Security Symposium, 2001

[29] Digital Equipment Corporation – *Performance tuning tips for Digital Unix*, iunie 1996, http://www.abdn.ac.uk/local/apache/manual_1.3.4/misc/perf-dec.html

[30] D. E. Denning, G. Sacco - *"Timestamps in Key Distribution Protocols"*, Communications of the ACM, Vol. 24, 1981

[31] Prashant Dewan, Partha Dasgupta, Vijay Karamcheti – *Defending Against Denial of Service Attacks Using Name Secure Resolution*, The 23rd International Conference on Distributed Computing, 2003

[32] The Distributed.net Organization, http://www.distributed.net

[33] Cynthia Dwork, Moni Naor – *Pricing via Processing or Combating Junk Mail*, Proceedings of CRYPTO '92, Springer Verlag, 1992

[34] William Enck, Patrick Traynor, Patrick McDaniel, Thomas la Porta, "Exploiting open functionality in SMS capable cellular networks", *12th ACM Conference on Computer and Communication Security*, 2005

[35] D. C. Feldmeier, P. R. Karn – *UNIX Password Security – Ten Years Later*, Advances in Cryptology – CRYPTO '89, Springer-Verlag, Berlin, 1990

[36] P. Ferguson, D. Senie – *Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing*, RFC 2267, 1998

[37] Alan O. Freier, Philip Karlton, Paul C. Kocher – *The SSL Protocol, Version 3.0 (Internet Draft)*, Transport Layer Security Working Group, 1996

[38] Emmanuel Gadaix – *GSM and 3G Security*, Black Hat Conference Singapore, 2001

[39] Ghosh and Swaminatha - *M-commerce Security*, Communications of the ACM, February 2001

[40] David M. Goldschlag, Michael G. Reed, Paul F. Syverson – *Onion routing for anonymous and private Internet connections*, Communications of the ACM, 42(2):84-88, January 1999

[41] L. Gong, R. Needham, R. Yahalom – *"GNY Logic Fill In"*, Proceedings of the IEEE Symposium on Security and Privacy, 1990

[42] L. Gong – *"A Security Risk of Depending on Synchronized Clocks"*, ACM Operating Systems Review, Vol. 26, 1992

[43] L. Gong, T. M. A. Lomas, R. M. Needham, J. H. Saltzer – *Protecting Poorly Chosen Secrets from Guessing Attacks*, IEEE Journal on Selected Areas in Communications, Vol. 11, June 1993

[44] B. Groza, D. Petrica – *On Chained Cryptographic Puzzles*, 3rd Romanian-Hungarian Joint Symposium on Applied Computational Intelligence (SACI), Timisoara, Romania, May 25-26 2006

[45] Gunnar Heine – *GSM Networks: Protocols, Terminology and Implementation*, Alcatel SEL Germany, 1998

[46] Michael Howard, David LeBlanc, *Writing Secure Code*, 2nd Edition, Microsoft Press, 2003

[47] Andrew Huang, "Keeping Secrets in Hardware: the Microsoft XBox™ Case Study, May 2002, http://web.mit.edu/bunnie/www/proj/anatak/AIM-2002-008.pdf

[48] Ari Juels, John Brainard – *Client puzzles: A cryptographic defense against connection depletion attacks*, Proceedings of the NDSS 1999

[49] Y. Kim, W. C. Lau, M. C. Chuah, H. J. Chao - *PacketScore: Statistics-based Overload Control against Distributed Denial-of-Service Attacks*, in *IEEE INFOCOM 2004*, Hong Kong, 2004.

[50] D. V. Klein – *"Foiling the Cracker": A Survey of, and Improvements to, Password Security,* Proceedings of the USENIX UNIX Security II Symposium, USENIX Association, Berkeley 1990

[51] J. Kohl – *The Kerberos Network Authentication Service (V5) (RFC 1510)*, Digital Equipment Corporation, 1993

[52] J. T. Kohl, B. C. Neuman, T. Y. Ts'o – *The Evolution of the Kerberos Authentication System*, Distributed Open Systems, IEEE Computer Society Press, Los Alamitos, CA, 1994

[53] K. Y. Lam, T. Beth – *Timely Authentication in Distributed Systems",* ESORICS '92 – European Symposium on Research in Computer Security, Springer-Verlag, Berlin, November 1992

[54] Liberty Alliance Project – *Liberty ID-FF Architecture Overview*, 2003, http://www.projectliberty.org

[55] Liberty Alliance Project – *Liberty ID-FF Protocols and Schema Specification 1.2*, 2003, http://www.projectliberty.org

[56] Liberty Alliance Project – *Liberty Specs Tutorial*, http://www.projectliberty.org

[57] T. M. A. Lomas, L. Gong, J. H. Saltzer, R. M. Needham – *Reducing Risks from Poorly Chosen Keys*, ACM Operating Systems Review, Vol. 23, 1989

[58] Steve Lord – *Bugwatch: GSM security flaws exposed*, VNU Business Publications Limited, 2003, http://www.vnunet.com/vnunet/news/2121449/bugwatch-gsm-security-flaws-exposed

[59] M. Ma – *Mitigating denial of service attacks with password puzzles*, International Conference on Information Technology: Coding and Computing, 2005, volume 2, pag. 621-626

[60] David Margrave – *GSM Security and Encryption*, George Mason University

[61] Catherine Meadows – *A formal framework and evaluation method for network denial of service*, Proceeding of the 1999 IEEE Computer Security Foundations Workshop, Mordano, Italy, 1999

[62] R. C. Merkle – *Secure Communications Over Insecure Channels*, Communications of the ACM, 1978

[63] S. K. Nair, B. C. Popescu, C. Gamage, B. Crispo and A. S. Tanenbaum - "*Enabling DRM – preserving Digital Content Redistribution*", 7th International IEEE Conference on E-Commerce Technology, 2005

[64] R. M. Needham, M. D. Schroeder – *"Using Encryption for Authentication in Large Networks of Computers"*, Communications of the ACM, Vol. 21, December 1978

[65] R. M. Needham, M. D. Schroeder – *"Authentication Revisited"*, ACM Operating Systems Review, Vol. 21, 1987

[66] Rolf Oppliger – *Authentication Systems for Secure Networks*, Artech House, Inc., 1996

[67] Andreas Pashalidis, Chris J. Mitchell – *A Taxonomy of Single Sign-On Systems*, 2003, Royal Holloway, University of London

[68] Andreas Pfitzmann, Marit Köhntopp – *Anoymity, unobservability and pseudonimity – a proposal for terminology*, Designing Privacy Enhancing Technologies, International Workshop on Design Issues in Anonymity and

Unobservability, number 2009 in Lecture Notes in Computer Science, pages 141 – 160, Springer-Verlag, Berlin 2001

[69] Ping-Herng Denny Lin – *Survey of the Denial of Service Countermeasures*, California State University, Fullerton, 2000

[70] J. J. Quisquater, L. Guillou – *"How to Explain Zero-Knowledge Protocols to Your Children"*, Advances in Crytoplogy – CRYPTO '89, Springer – Verlag, Berlin 1990

[71] Valentin Razmov – *Denial of Service Attacks and How to Defend Against Them*, University of Washington, 2000

[72] Ronald R. Rivest, Adi Shamir, David A. Wagner – *Time-lock Puzzles and Timed-release Cryptography*, 1996, http://lcs.mit.edu/~rivest/RivestShamirWagner-timelock.pdf

[73] Stefan Savage, David Wetherall, Anna Karlin, Tom Anderson – *Practical network support for IP traceback*, technical report UW-CSE-00/02/0, SIGCOMM '00, 2000

[74] Bruce Schneier – *Distributed denial of service attacks*, Crypto-gram newsletter, 2000

[75] Schneier, Bruce - *Secrets & Lies,* Wiley Publishing, Inc., 2004

[76] Bruce Schneier – *Applied Cryptography*, John Wiley & Sons, Inc., 1996

[77] SETI @home Program, http://setiathome.ssl.berkely.edu

[78] Shibboleth Architecture, Protocols and Profiles, Working Draft 02, 22 September 2004, http://shibboleth.internet2.edu

[79] C. Siaterlis and B. Maglaris - *Towards Multisensor DataFusion for DoS Detection, in SAC'04*, Nicosia, Cyprus, 2004

[80] Oliver Spatscheck, Larry Peterson – *Defending against denial of service in Scout*, Proceedings of 3rd USENIX/ACM Symposium on OSDI, p. 59-72, 1999

[81] William Stallings - *Cryptography and Network Security, Principles and Practices*, Third Edition, Prentice Hall, 2003

[82] J. G. Steiner, B. C. Neuman, J. I. Schiller – *Kerberos: An Authentication Service for Open Network Systems*, Proceedings of the USENIX UNIX Security Symposium, 1988

[83] Filip Šuba – *Usability and Security of DRM architectures*, TKK T-110.5190 Seminar on Internetworking, Helsinki University of Technology, 2007

[84] T. Aura, P. Nikander, J. Leiwo – *DOS-resistant authentication with client-puzzles*, Proceeding of the Cambridge Security Protocols Workshop 2000, LNCS, Cambridge, UK, 2000

[85] Upkar Varshney – *Network access and security issues in ubiquitous computing*, Workshop on Ubiquitous Computing Environment, Cleveland, 2003

[86] B. Waters, A. Juels, J. A. Halderman, E. W. Felten – *New Client Puzzle Outsourcing Techniques for DoS Resistance*, 11th ACM Conference on Computer and Communications Security, 2004

[87] X. Wang, M. K. Reiter – *Defending Against Denial-of-Service Attacks with Puzzle Auctions (Extended Abstract)*, 2003 IEEE Symposium on Security and Privacy (SP'03), pages 78-92

[88] IBM Corporation, Microsoft Corporation, BEA Systems, RSA Security, Verisifg - *Web Services Federation Language (WS-Federation)*, July 2003, http://www.ibm.com/developerworks/library/ws-fed/

[89] Anonimizer – www.anonymizer.com

[90] WebSecure - http://www.freedom.net/products/websecure

[91] JAP Anonymity and Privacy – http://anon.inf.tu-dresden.de

[92] Trusted Computing Group, "Trusted Computing Platform Alliance Main Specification", October 2003, Version 1.2, http://www.trustedcomputinggroup.org

[93] Eric W. Weisstein, "Chinese Remainder Theorem." From *MathWorld* - A Wolfram Web Resource, http://mathworld.wolfram.com/ChineseRemainderTheorem.html

[94] eXtensible Access Control Markup Language (XACML), http://www.oasis-open.org/committees/xacml

[95] XrML: eXtensible Rights Markup Language, http://www.xrml.org

[96] Content Scrambling System, http://www.dvdcca.org/css/

[97] Overview of the Protected Media Path, http://msdn2.microsoft.com/en-gb/library/aa376846.aspx

[98] Mentalis Secure Socket Library, http://www.mentalis.org/soft/projects/ssocket/

[99] BigInteger Project, http://www.codeproject.com/csharp/biginteger.asp

[100] World Wide Web Consortium – *The Platform for Privacy Preferences 1.0 (P3P 1.0) Specification, April 2002*

[101] A. Fiat, A. Shamir – "How to Prove Yourself: Practical Solutions to Identification and Signature Problems", Advances in Crytoplogy – CRYPTO '86, Springer – Verlag, Berlin 1987

[102] Shon Harris – DOS Defense, Information Security Magazine, 2001

[103] Valer BOCAN, Vladimir CREŢU - *Security and Denial of Service Threats in GSM Networks*, CONTI 2004 - Periodica Politehnica, Transaction on Automatic Control and Computer Science Vol. 49 (63), 2004, ISSN 1224-600X

[104] Valer BOCAN - *Sisteme Single Sign-On sub atacuri Denial-of-Service. Studiu de caz: Proiectul Liberty Alliance*, Referat de doctorat nr. 3, Politehnica University of Timisoara

[105] Valer BOCAN - *Studiu asupra nivelului de siguranta oferit de protocoalele de autentificare*, Raport de doctorat nr. 2, Politehnica University of Timisoara, 2004

[106] Valer BOCAN - *Stadiul actual al dezvoltarii sistemelor de securitate pentru retele de calculatoare de inalta siguranta*, Raport de doctorat nr. 1, Politehnica University of Timisoara

Lista publicaţiilor personale

2008

Valer BOCAN, Vladimir CREŢU - *SCADDIC: The Implementation and Performance of a Scalable and Secure Architecture for Digital Content Distribution*, 7th Conference on Communications, Military Technical Academy, Bucharest, 2008

2006

Valer BOCAN, Vladimir CREŢU - *Threats and Countermeasures in GSM Networks*, Journal of Networks, Academy Publishers, ISSN 1796-2056

Valer BOCAN, Mihai FĂGĂDAR-COSMA - *Scalable and Secure Architecture for Digital Content Distribution*, SoftCOM 2006 - International Conference on Software, Telecommunications and Computer Networks, Split, Croatia

Valer BOCAN, Vladimir CREŢU - *Mitigating Denial of Service Threats in GSM Networks*, ARES 2006 - The First International Conference on Availability, Reliability and Security, Wien, Austria

2005

Valer BOCAN, Mihai FĂGĂDAR-COSMA - *Adaptive Threshold Puzzles,* EUROCON 2005 - The International Conference on "Computer as a tool", Belgrade, Serbia and Montenegro

Valer BOCAN, Mihai FĂGĂDAR-COSMA - *Towards DoS-resistant Single Sign-On Systems*, EUROCON 2005 - The International Conference on "Computer as a tool", Belgrade, Serbia and Montenegro

Valer BOCAN - *Sisteme Single Sign-On sub atacuri Denial-of-Service. Studiu de caz: Proiectul Liberty Alliance*, Referat de doctorat nr. 3, Politehnica University of Timisoara

2004

Valer BOCAN - *Studiu asupra nivelului de siguranta oferit de protocoalele de autentificare*, Raport de doctorat nr. 2, Politehnica University of Timisoara, 2004

Valer BOCAN - *Developments in DoS Research and Mitigating Technologies,* CONTI 2004 - Periodica Politehnica, Transaction on Automatic Control and Computer Science Vol. 49 (63), 2004, ISSN 1224-600X

Valer BOCAN, Vladimir CREŢU - *Security and Denial of Service Threats in GSM Networks*, CONTI 2004 - Periodica Politehnica, Transaction on Automatic Control and Computer Science Vol. 49 (63), 2004, ISSN 1224-600X

Valer BOCAN - *Threshold Puzzles. The Evolution of DoS-Resistant Authentication*, CONTI 2004 - Periodica Politehnica, Transaction on Automatic Control and Computer Science Vol. 49 (63), 2004, ISSN 1224-600X

2002

Valer BOCAN - *Stadiul actual al dezvoltarii sistemelor de securitate pentru retele de calculatoare de inalta siguranta*, Raport de doctorat nr. 1, Politehnica University of Timisoara