UNIUNEA EUROPEANĂ
GUVERNUL ROMÂNIEI MINISTERUL MUNCII, FAMILIEI ŞI PROTECŢIEI SOCIALE AMPOSDRU
Fondul Social European POSDRU 2007-2013
Instrumente Structurale 2007-2013
ORGANISMUL INTERMEDIAR REGIONAL PENTRU POS DRU REGIUNEA BUCUREŞTI ILFOV

Proiect cofinanțat din Fondul Social European prin
Programul Operațional Sectorial Dezvoltarea Resurselor Umane 2007-2013.
"Investește în oameni!"

# *Cap. 5.3. Protocoale de autentificare in retele de calculatoare (EKE, STS, IPSec, SSL/TLS, SSH, Kerkeros).*

# Kerberos

- Dezvoltat la MIT ca parte a proiectului Athena, folosit in special in Windows, dar si pe Unix
- Ideea de baza: utilizarea unei parti de incredere pentru distributie de chei de sesiune ce pot fi utilizate intre utilizator si diverse servere
- Compus din 3 sub-protocoale:
  - Authentication Service Exchange (AS) – ruleaza intre C si AS
  - Ticket-Granting Service Exchange (TGS) – ruleaza intre C si TGS
  - Client/Server Authentication Application Exchange (AP) – ruleaza intre C si AP
- Diviziunea intre AS si TGS este utila pentru cazul in care serverele de aplicatie apartin la domenii diferite (deci U poate fi deservit de un singur AS dar de mai multe TGS)

# Participanti la protocol

- U – utilizator al serviciului de single-sign-on (SSO)
- C – aplicatie client invocata de user
- S – server de aplicatie (care detine resursele necesare aplicatiei)
- KDC – Key Distribution Center format din 2 componente:
  - AS – Authentication Server – serverul la care C se autentifica (in faza intiala folosind o parola fixata prin alt mijloc decat Kerberos)
  - TGS – Ticket Grantig Server – elibereaza "bilete" pe care C le foloseste pentru a se autentifica la S

- Diviziunea intre AS si TGS este utila pentru cazul in care serverele de aplicatie apartin la domenii diferite (deci U poate fi deservit de un singur AS dar de mai multe TGS)

# Protocolul Kerberos

$$1.C \rightarrow AS : U, TGS, LifeTime, N_1$$

$$2.AS \rightarrow C : U, T_{C,TGS} = Enc_{K_{AS,TGS}} \{U, C, TGS, K_{C,TGS}, TimeStart, TimeExp\},$$

$$TGT_C = Enc_{K_U} \{TGS, K_{C,TGS}, TimeStart, TimeExp, N_1\}$$

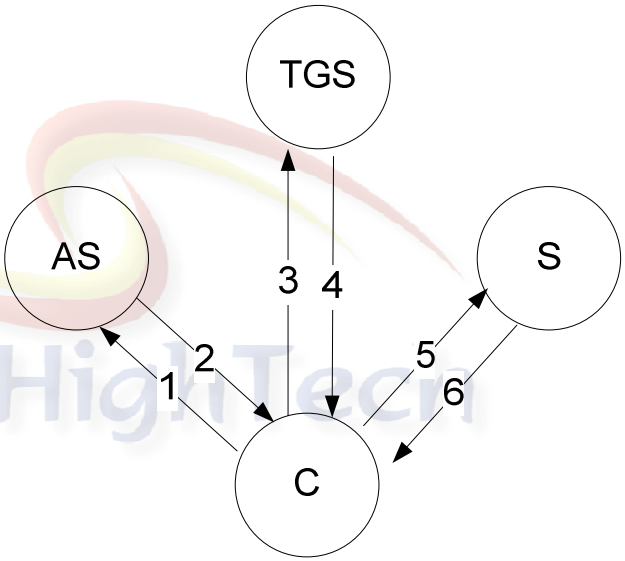
$$3.C \rightarrow TGS : S, LifeTime, N_2, T_{C,TGS}, A_{C,TGS} = \{C, ClientTime\}_{K_{C,TGS}}$$

$$4.TGS \rightarrow C : U, T_{C,S} = E_{K_{S,TGS}} \{U, C, S, K_{C,S}, TimeStart, TimeExp\},$$

$$TKT_C = E_{K_{C,TGS}} \{S, K_{C,S}, TimeStart, TimeExp, N_2\}$$

$$5.C \rightarrow S : T_{C,S}, A_{C,S} = E_{K_{C,S}} \{C, ClientTime\}$$

$$6.S \rightarrow C : A_{S,C} = E_{K_{C,S}} \{ClientTime\}$$



# Station to station protocol (STS)

- Propus de Diffie, Oorschot si Wiener in 1992 (nume grele)
- Si are cinci proprietati grele de securitate:
  - Autentificare mutuala
  - Schimb mutual de cheie
  - Confirmare mutuala a cheii
  - Perfect forward secrecy – chiar daca o cheie privata este compromisa, nu pot fi aflate chei anterior folosite
  - Anonimitate – un observator din retea nu poate demonstra ca e vorba de comunicare intre 2 participanti anume

$$1.A \rightarrow B : \alpha^x$$

$$2.B \rightarrow A : \alpha^y, Cert_B, E_k\left(sig_B\left(\alpha^y, \alpha^x\right)\right)$$

$$3.A \rightarrow B : Cert_A, E_k\left(sig_B\left(\alpha^x, \alpha^y\right)\right)$$

# Propus alaturi de varianta STS "Authentication only"

- Autorii sustin ca este in esenta ISO Public Key Three-Pass Mutual Authentication

$$1.A \rightarrow B : N_A$$
$$2.B \rightarrow A : Cert_B, N_B, sig_B(N_B, N_A)$$
$$3.A \rightarrow B : Cert_A, sig_A(N_A, N_B)$$

- Din pacate are o scapare esentiala: identitatile participantilor nu sunt semnate
- In atacul de mai jos: A vorbeste cu Adv iar B crede ca vorbeste cu A dar de fapt vorbeste tot cu Adv

$$1.A \rightarrow Adv : N_A$$
$$2.Adv(A) \rightarrow B : N_A$$
$$3.B \rightarrow Adv(A) : Cert_B, N_B, sig_B(N_A, N_B)$$
$$4.Adv \rightarrow A : Cert_B, N_B, sig_B(N_A, N_B)$$
$$5.A \rightarrow Adv : Cert_A, sig_A(N_A, N_B)$$
$$6.Adv(A) \rightarrow B : Cert_A, sig_A(N_A, N_B)$$

# Atacul lui Lowe'94 asupra STS

- Considerat un atac minor, are urmatorul rezultat A crede ca vorbeste B, B crede ca vorbeste cu Adv intr-o sesiune incompleta
- Numit si atac DoS deoarece pentru A comunicatia post-autentificare nu continua (Adv tot nu stie cheia secreta) cu toate ca protocolul se incheie corect

$$1.Adv(B) \rightarrow A : start$$

$$2.A \rightarrow Adv(B) : \alpha^x$$

$$3.Adv \rightarrow B : \alpha^x$$

$$4.B \rightarrow Adv : \alpha^y, Cert_B, E_K\left(sig_B\left(\alpha^y, \alpha^x\right)\right)$$

$$5.Adv(B) \rightarrow A : \alpha^y, Cert_B, E_K\left(sig_B\left(\alpha^y, \alpha^x\right)\right)$$

$$6.A \rightarrow Adv(B) : Cert_A, E_K\left(sig_A\left(\alpha^x, \alpha^y\right)\right)$$

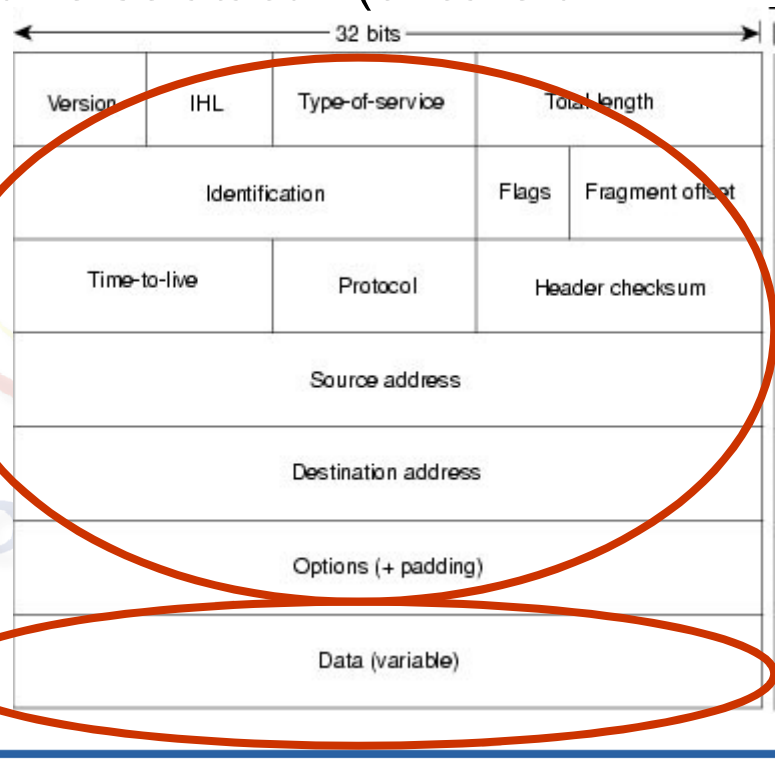
*Proiect cofinanțat din Fondul Social European prin Programul Operațional Sectorial Dezvoltarea Resurselor Umane 2007-2013. "Investește în oameni!"*

# Internet Security - IPSec

- Obiectiv: **obligatoriu autentificare** si **optional confidetialitate**
- Se doreste asigurarea protectiei pentru un header IP
- Motivatie: In absenta protectiei sunt posibile diverse atacuri (oricare din atacurile mentionate, de. ex. reflectie)

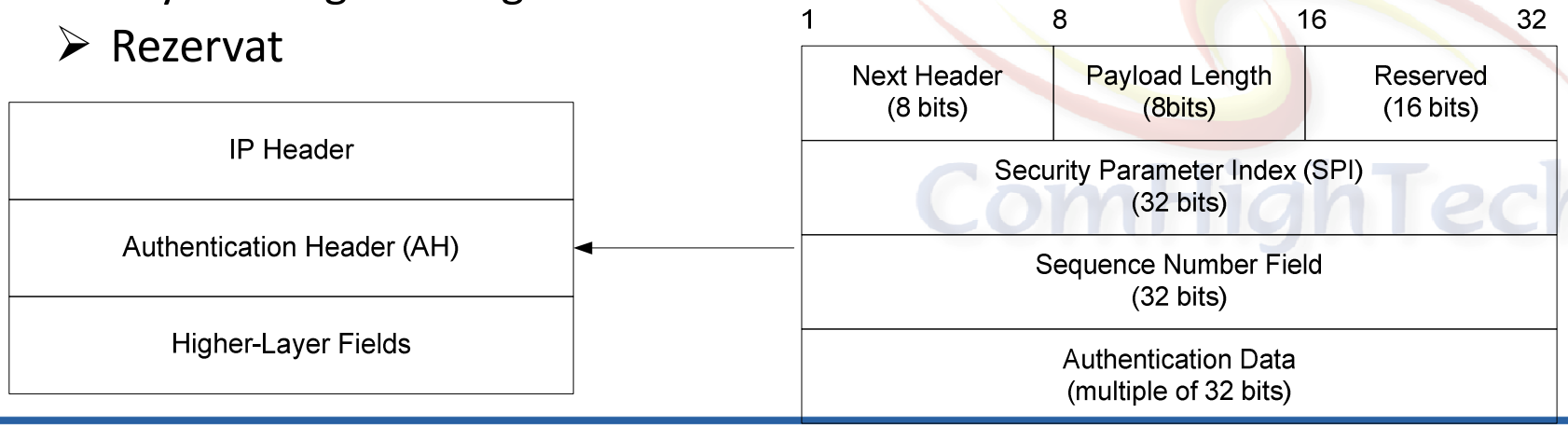
header IP

Campuri pentru layerul de nivel inalt (7)



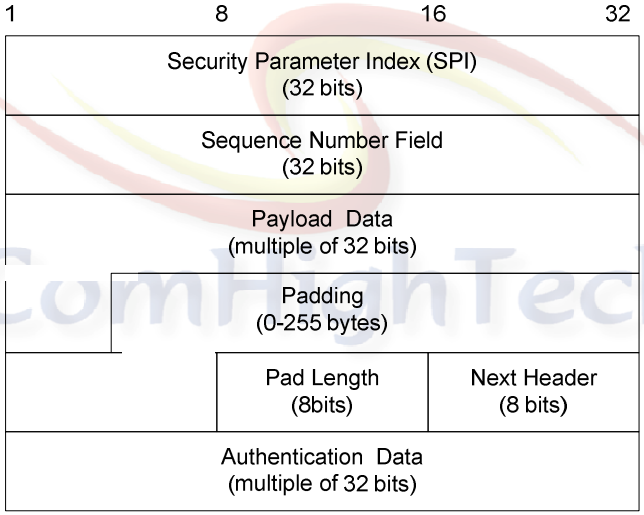
# Asigurarea autenticitatii in IPSec

- Introducerea unui Authentication Header (AH)
- Campuri pentru securitate:
  - Security Parameter Index (SPI) – specifica modul de functionare (tunel/transport) si algoritmii utilizati (HMAC-SHA1, 3DES, AES)
  - Sequence Number – contor pentru a evita atacuri de tip replay
- Campuri fara semnificatie in securitate:
  - Next Header – tipul urmatorului header
  - Payload Legth – lungimea headerului
  - Rezervat



# Asigurarea confidentialitatii in IPSec

- Blocuri de 32 de biti numite ESP (Encapsulating Security Payload) este alocata dupa un AH
- Campurile au aceeasi semnificatie
- Payload Data sunt datele criptate si padding este folosit pentru ca lungimea sa fie multiplu de 32 biti
- Authentication Data este acum AH si este optional (greseala fatala!)



# Internet Key Exchange (IKE)

- Necesar pentru a stabili cheia in IPSec (pentru a stabili relatia de Security Association (SA) intre doua noduri)
- IKE este o suita de protocoale de schimb de cheie autentificat, majoritatea bazate pe Diffie-Hellman
- IKE este compus din 2 faze (Phase1 + Phase2)
- Faza 1 are 8 variante este un schimb de cheie cu autentificare mutuala
- Faza 2 (denumita si quick mode) poate avea loc de mai multe ori dupa faza 1si poate fi folosita pentru a stabili mai multe tipuri de conexiuni cu diferiti parametrii de securitate: integrity only, encryption only etc.

*Proiect cofinanțat din Fondul Social European prin Programul Operațional Sectorial Dezvoltarea Resurselor Umane 2007-2013.*
*"Investește în oameni!"*

# IKE – Faza 1

- Exista 8 moduri de faza 1 deoarece sunt 4 tipuri de chei (pre-shared symmetric, old-style public key, new-style public key si public signature verification key) si 2 tipuri de schimb (main mode si aggressive mode)
- Modul main schimba 6 mesaje (3+3) si este obligatoriu
- Modul aggressive schimba 3 mesaje si este optional

# Signature-based IKE Phase 1 Main Mode

$$1.A \rightarrow B : HDR_A, SA_A$$
$$2.B \rightarrow A : HDR_B, SA_B$$
$$3.A \rightarrow B : HDR_A, g^x, SA_A$$
$$4.B \rightarrow A : HDR_B, g^y, SA_B$$
$$5.A \rightarrow B : HDR_A, E_{g^{xy}}\left(ID_A, Cert_A, Sig_A\right)$$
$$6.B \rightarrow A : HDR_B, E_{g^{xy}}\left(ID_B, Cert_B, Sig_B\right)$$

- Inrudit cu protocolul STS, difera prin criptarea certificatelor (utilizata pentru anonimitate) si semnarea cheii stabilite

$$Sig_A = Sig_A\left(M_1\right)$$
$$M_1 = prf_1\left(prf_2\left(N_A \| N_B \| g^{xy}\right) \| g^x \| g^y \| C_A \| C_B \| SA_A \| ID_A\right)$$
$$Sig_B = Sig_B\left(M_2\right)$$
$$M_2 = prf_1\left(prf_2\left(N_A \| N_B \| g^{xy}\right) \| g^x \| g^y \| C_B \| C_A \| SA_B \| ID_B\right)$$

HDR – headere care contin cookie C

SA – security association care negociaza parametrii de securitate: algoritmi de criptare, semnare, hash etc. (se pot propune mai multe variante si se raspunde cu alegerea uneia)

ID – identitatile participantilor

# Atac similar atacului lui Lowe

- Atacul functioneaza in primul rand pentru ca mesajul semnat contine doar identitatea celui care semneaza, nu si cea a presupusului partener
- In urma atacului B crede ca vorbeste cu A, in timp ce A crede ca vorbeste cu Adv intr-o sesiune neterminata

$$1.A \rightarrow Adv : HDR_A, SA_A$$
$$1'.Adv(A) \rightarrow B : HDR_A, SA_A$$
$$2'.B \rightarrow Adv(A) : HDR_B, SA_B$$
$$2.Adv \rightarrow A : HDR_B, SA_B$$
$$3.A \rightarrow Adv : HDR_A, g^x, N_A$$
$$3'.Adv(A) \rightarrow B : HDR_A, g^x, N_A$$
$$4'.Adv(B) \rightarrow A : HDR_B, g^y, N_B$$
$$4.Adv \rightarrow A : HDR_B, g^y, SA_B$$
$$5.A \rightarrow Adv : HDR_A, E_{g^{xy}}(ID_A, Cert_A, Sig_A)$$
$$5'.Adv(A) \rightarrow B : HDR_A, E_{g^{xy}}(ID_A, Cert_A, Sig_A)$$
$$6.'B \rightarrow Adv(A) : HDR_B, E_{g^{xy}}(ID_B, Cert_B, Sig_B)$$
$$6.Abandon$$

# IKE – Aggressive Mode

- Simplificare a modului de baza, prin reducere de la 6 la 3 mesaje
- Un atac similar cu cel al lui Lowe este fezabil

$$1.A \rightarrow B : HDR_A, SA_A, g^x, N_A, ID_A$$

$$2.B \rightarrow A : HDR_B, SA_B, g^y, N_B, ID_B, Cert_B, Sig_B$$

$$3.A \rightarrow B : HDR_A, Cert_A, Sig_A$$

# Probleme ale IPSec si IKE

- Permite atacuri DoS prin faptul ca un participant poate fi determinat sa desfasoare operatii criptografice intense fara a avea vreo garantie cu privire la partenerul de comunicare
- Complexitatea ridicata a protocolului (multe variante si flexibilitate) nu este un factor pozitiv de securitate
- Schneier si Ferguson critica complexitatea ca efect de comitet
- IKE v2 repara parte din probleme lui IKE
- IKE v2 ofera anonimitate (plausible deniability), obiectiv dorit in diverse servicii IP

# Secure Shell (SSH) remote login protocol

- Secure Shell (SSH) protocol de autentificare bazat pe cheie publica intre un server si un client
- Standard pe sistemele UNIX, exista si pentru Win
- Obiectiv: crearea unui canal sigur (autentic si confidential) pentru executia de comenzi, mutarea de fisiere etc.
- Trei componente majore:
  - SSH Transport Layer – autentifica serverul catre client si creeaza un canal sigur intre client si server (exista 2 strategii posibile: clientul are o baza de date cu chei publice ale serverelor de incredere sau clientul cunoaste autoritatea care a semnat certificatul serverului)
  - SSH User Authentication Protocol – autentifica clientul catre server (folosind chei publice sau mai uzual parole)
  - SSH Connection Protocol – este un tunel criptat prin care se trimit comenzile

# Schimbul de cheie SSH

- Semnificatii deja cunoscute ale valorilor
- $V_C$, $V_S$ – versiunea protocolului
- $I_C$, $I_S$ – valori intiale (schimbate inainte de a se ajunge la pasul curent)

$$1.C \rightarrow S : e = g^x \bmod p$$

$$2.S \rightarrow C : K_S \,||\, f = g^y \bmod p \,||\, s = Sig_S\left(H\left(V_C \,||\, V_S \,||\, I_C \,||\, I_S \,||\, K_S \,||\, e \,||\, f \,||\, K\right)\right)$$

$$3.C \rightarrow S : verificare : K = f^x, Ver_S\left(s\right)$$

- Daca verificare de la pasul 3 se incheie cu succes, se trece la autentificarea clientului (uzual folosind parole) pe un canal sigur (criptat)

# SSL si TLS

- Protocol de autentificare unilaterala (de cele mai multe ori) sau mutuala destinat in special pentru WorldWideWeb
- Destinat socketurilor de aici numele de SecureSocketLayer
- Dezvoltat initial de Netscape, si ulterior acceptat ca standard de toti dezvoltatorii inclusiv Microsoft
- Evolueaza in Transport Layer Security (TLS) sub corpul de standardizare Internet Engineering Task Force (IETF)
- Ruleaza sub nivelul aplicatie (HTTP, IMAP etc.) si deasupra nivelelor retea TCP/IP
- Este format din 2 componente
  - TLS Record Protocol – calculeaza MAC-uri (integritate/autenticitate), cripteaza (confidentialitate), optional compreseaza, si la receptie decripteaza/verifica
  - TLS Handshake Protocol – componenta de autentificare, negociere a algoritmilor si a cheii

# Structura unui handshake TLS

- Mesajele Hello au ca scop stabilirea valorilor: versiune protocol, random, id sesiune, algoritmi de criptare, compresie
- Certificatele sunt in formatul X.509
- KeyExchange contine partea de cheie Diffie-Hellman (dar functioneaza si cu schimb de cheie dinspre client folosind RSA)
- Valorile marcate cu $^1$ sunt optionale

$$1.C \rightarrow S : ClientHello$$

$$2.S \rightarrow C : ServerHello, ServerCertificate^1, ServerKeyExachange^1, CertificateRequest^1, ServerHelloDone$$

$$3.C \rightarrow S : ClientCertificate^1, ClientKeyExchange, CertificateVerify^1, ClientFinished$$

$$4.S \rightarrow C : ServerFinished$$

- Uzual clientul ramane anonim si nu trimite certificat

*Proiect cofinanțat din Fondul Social European prin Programul Operațional Sectorial Dezvoltarea Resurselor Umane 2007-2013.*
*"Investește în oameni!"*

# Securitatea SSH si SSL/TLS

- Ofera un nivel suficient de bun de securitate pentru majoritatea utilizatorilor
- Nu au vulnerabilitati semnificative (variantele noi si corect implementate)
- Sunt susceptibile la atacuri prin temporizare (side-channel)