Kriptográfia és Információbiztonság
6_szt. előadás

MÁRTON Gyöngyvér

Sapientia Egyetem, Matematika-Informatika Tanszék
Marosvásárhely, Románia
mgyongyi@ms.sapientia.ro

2023

IP biztonság - IP security, IPsec

- van amikor olyan biztonsági megoldásokra van szükség, amely több réteget is érint, mert nem elég a különböző alkalmazások által nyújtott biztonság
- egy privát IP-hálózatot üzemeltetése:
  - letilthatók a nem megbízható oldalakra mutató hivatkozások
  - titkosíthatók a belső hálózatról kimenő tartalmak
  - hitelesíthető a külső hálózatról érkező adatforgalom
- az IP-szintű biztonság olyan alkalmazások biztonságos használatát is lehetővé teszi, amelyek egyébként nem rendelkeznek ilyen funkcióval
- előnye: átlátható az alkalmazások számára
- hátránya:
  - összetett, túltervezett, funkciói nem egyértelműek, nehéz implementálni, több komoly biztonsági hibája van
  - a specifikáció három részre van felosztva: RFC 2407, RFC 2408, RFC 2409, amelyeket különböző, egymástól független módon, egymástól független szervek írtak le

IP biztonság - IP security, IPsec

IPsec SSL/TLS különbségek:

- az SSL/TLS a felhasználói rétegben, míg az IPsec a hálózati rétegben, az operációs rendszer részeként van implementálva
- mindkét protokoll biztosítja a hitelesítést, kulcscserét bizalmas adatcserét
- az SSL/TLS egyszerű, jól megtervezett, az IPsec bonyolult, hibát is tartalmaz
- ha módosítani kell az SSL/TLS-t az nem vonja maga után az operációs rendszer módosítását, ellentétben az IPsec-el, amelyet ha módosítani kell, akkor az operációs rendszert is módosítani kell
- az alkalmazások esetében fordítva van: ha módosítani kell az SSL/TLS-t akkor az alkalmazásokat is módosítani kell, az IPsec esetében azonban nem
- az SSL/TLS elsődleges használata: a webes tranzakciók felügyelete
- IPsec elsődleges használata: a VPN (virtual private network) biztonságának szavatolása, különböző hálózati végpontok közötti biztonságos kommunikáció megvalósítása

IP biztonság - IP security, IPsec

Alkalmazási területei:

- belső virtuális hálózatot építhető: a vállalkozások nem kell költséges magánhálózatot kiépítsenek
- távoli hozzáférés: egy felhasználó hozzáférést kaphat egy vállalat belső hálózatához
- biztonságos kommunikáció más egységekkel: biztosítja a hitelesítést, a kulcscserét és bizalmas információcserét,
- elektronikus kereskedelem: annak ellenére, hogy a kereskedelmi alkalmazások rendelkeznek beépített biztonsági protokollokkal, az IPsec használata fokozza ezt a biztonságot, garantálja, hogy a hálózaton keresztül haladó adatforgalom titkosítva és hitelesítve legyen

Jellemzői:

- felügyeli az összes elosztott alkalmazás biztonságát: a távoli bejelentkezést, a klienst/szerver kapcsolatot, az e-mail szolgáltatást, a fájlátvitelt, a webkapcsolatokat, stb.
- az IPsec routerekbe, tűzfalakba van beépítve, azaz olyan eszközökbe, amelyek összekötik a LAN-okat (Local Area Network) a külvilággal
- az IPsec eszközök titkosítják a WAN-ra (wide area network) kimenő forgalmat, és visszafejtik a WAN-ról érkező forgalmat

IP biztonság - IP security, IPsec

Két fontosabb része van:

- a kulcscsere - az IKE (InternetKey Exchange): biztosítja a kölcsönös hitelesítést és egy megosztott szimmetrikus kulcs létrehozását (ez a komplexebb)
- ESP/AH:
  - a beágyazott biztonságos kiegészítő - az ESP (Encapsulating Security Payload): biztosítja az IP-csomagok titkosítását, vagy integritását
  - a hitelesítő fejléc - az AH (Authentication Header): biztosítja az integritást
  - két módban alkalmazhatók
    - szállítási mód (transport mode)
    - alagút mód (tunel mode)

IP biztonság - az ESP és az AH

ESP és AH szállítási mód (transport mode):

- a felsőbb rétegben található protokollok számára nyújt védelmet
- a védelem az IP csomagok hasznos adat (payload) részére terjed ki: például a TCP- vagy UDP-szegmensre, vagy egy ICMP-csomagra (a hasznos adat az általában az IP-fejlécet követő adat)
- két egység(host) végpontok közötti (end-to-end) kommunikációjakor használják
- az ESP titkosítja és opcionálisan hitelesíti az IP hasznos adatot, de nem az IP fejlécet
- az AH hitelesíti az IP hasznos adatot és az IP-fejléc kiválasztott részeit

IP biztonság - az ESP és az AH

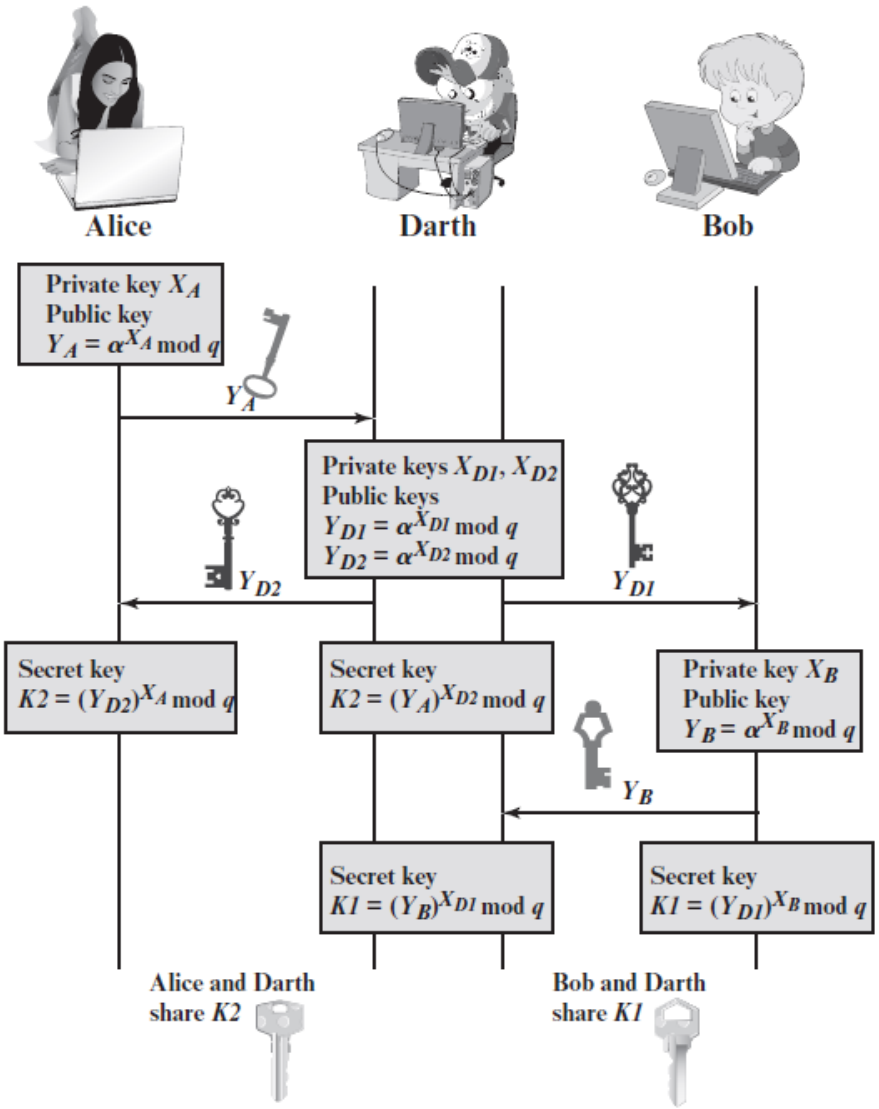
ESP és AH alagút mód (tunel mode):

- biztosítja a teljes IP csomag védelmét
- miután az AH vagy ESP mezőket hozzáadták az IP-csomaghoz, a teljes csomagot és a biztonsági mezőket a rendszer egy új külső IP-csomag hasznos terheként kezeli, új külső IP-fejléccel, az eredeti csomag tehát be van zárva
- az eredeti, belső csomag egy alagúton halad át az IP-hálózat egyik pontjától a másikig; a belső IP fejlécet nem lesz meghatározni: az új, nagyobb csomagnak teljesen eltérő forrás- és célcíme lehet
- alagút módban a tűzfalak mögötti hálózatokon számos egység IPsec nélkül is biztonságos kommunikációt folytathat
- az ESP titkosítja és opcionálisan hitelesíti a teljes belső IP-csomagot, beleértve a belső IP-fejlécet is
- az AH hitelesíti a teljes belső IP-csomagot és a külső IP-fejléc kiválasztott részeit

IP biztonság - az IKE, InternetKey Exchange

- a szimmetrikus/titkos kulcsok meghatározása és elosztása
- követelmény: az adatintegritás és a bizalmasság biztosítása érdekében különböző kulcs párok alkalmazása úgy a fogadó, mint a küldő oldalon,
- két kezelési mód:
  - manuális: a rendszergazda manuálisan konfigurálja a kulcsokat, kis, statikus környezetekben megvalósítható,
  - automatizált: nagy elosztott rendszerekben.

Man-in-the-Middle támadás



- minden kulcscsere protokoll támadható Man-in-the-Middle módszerrel, ha a protokoll résztvevői nem hitelesítik egymást
- kivédhető ha digitális aláírást alkalmaznak, ha publikus-kulcsokra vonatkozó tanúsítványokat használnak
- az ábra a Diffie-Hellman kulcsere protokoll elleni Man-in-the-Middle támadást mutatja be
- Darth a támadó, a támadást azzal indítja, hogy meghatározza a mindkét fél irányába szükséges privát és publikus értékeket (kulcsokat)

HTTPS

- a HTTPS (HTTP over SSL) a HTTP és az SSL kombinációjára utal, nem egy önálló protokoll, a HTTP felett fut és a biztonsági megoldásokra a TLS/SSL szabványt használja
- a felhasználó számára biztosítja a kommunikáció titkosságát, integritását és hogy a felkeresett weboldal hiteles, pl. védett a kommunikáció a *man-in-the-middle* típusú támadásokkal szemben
- a webszerver hitelesítését harmadik fél által kibocsátott tanúsítványok használatával oldja meg
- a biztonságos kommunikáció azonban csak akkor valósul meg, ha megfelelő titkosítási csomagok kerültek kiválasztásra, illetve megtörtént a webszerver hitelességének ellenőrzése
- minden modern webböngészőbe be van építve, használata webszervertől függ: egyes keresőmotorok például nem támogatják a HTTPS-t
- 2016 óta használják egyre szélesebb körben, dokumentációja az RFC2818-ben található

A HTTP és HTTPS közötti különbségek

HTTP:

- az URL-cím (uniform resource locator - egységes erőforrás-kereső): `http://`-vel kezdődik
- a 80-as portot használja
- az OSI modell szerint az alkalmazási rétegben helyezkedik el
- nem használ titkosítást, hitelesítést
- sebezhető a *man-in-the-middle* típusú támadással
- a biztonsági beállítások hiánya miatt gyorsabb
- információ megosztásra használt weboldalak esetében használják
- fenntartása nem jár költségekkel

HTTPS:

- az URL-cím: `https://`-vel kezdődik
- a 443-as portot használja
- az OSI modell szerint a szállítási rétegben helyezkedik el
- bizalmas információ cserét, integritást, hitelesítést biztosít
- nem támadható *man-in-the-middle* típusú támadással
- az alkalmazott biztonsági beállítások miatt lassúbb
- *érzékeny* adatokat (jelszavak, kártya adatok, stb) kezelő weboldalak esetében kell használni,
- fenntartása költségekkel jár: az tanúsítványokért fizetni kell

HTTPS

HTTPS esetén a következő elemek vannak titkosítva:

- a kért dokumentum URL címe
- a dokumentum tartalma
- a böngésző űrlapok tartalma (a böngésző felhasználója tölti ki)
- a böngészőről a szerverre és a szerverről a böngészőre küldött cookie-k
- a HTTP-fejléc tartalma

Akkor lehet megbízni egy weboldalban, amelyet HTTPS kapcsolaton keresztül értünk el, ha a következőkben is megbízunk:

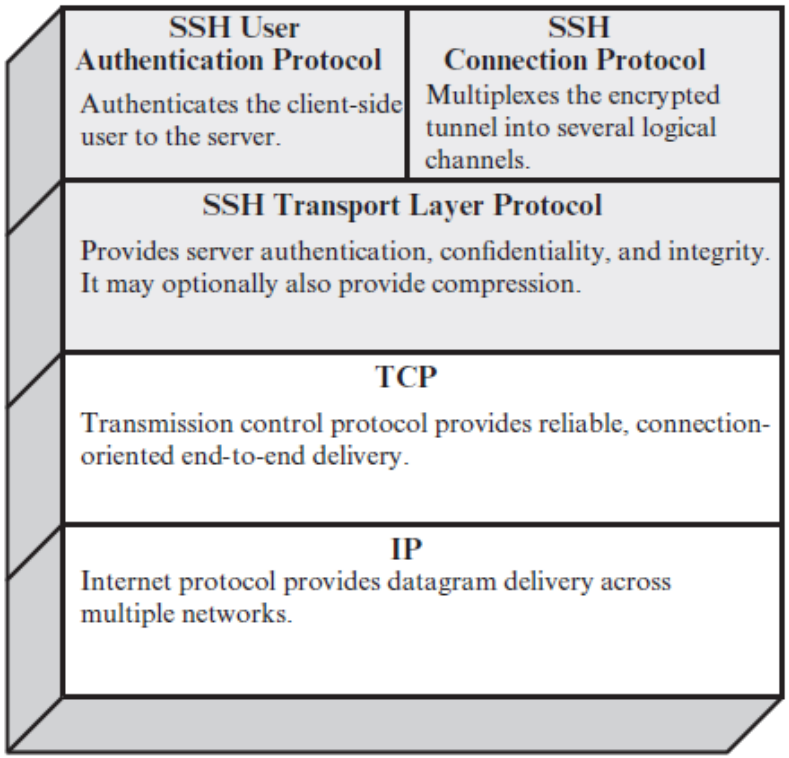
- a böngésző helyesen implementálta a HTTPS-t, illetve az előre telepített tanúsító hatóságokat
- a tanúsító hatóság csak legális webhelyeket garantál
- a weboldal érvényes tanúsítvánnyal rendelkezik
- a tanúsítvány helyesen azonosítja a weboldalt
- a titkosító algoritmusok elég *erősek*

Secure Shell-SSH

- biztonságos hálózati kommunikációra tervezett kliens-szerver architektúrán alapuló protokoll
- leggyakrabban biztonságos távoli parancssor, bejelentkezések, fájlátvitelek, emailek kezelését teszi lehetővé, viszonylag egyszerű és olcsó az implementálása
- a legtöbb operációs rendszerben elérhető
- megjelenése után, gyorsan a legelterjedtebb alkalmazássá vált a beágyazott rendszereken kívül
- új verziójában, az SSH2-ben, számos biztonsági hibát javítottak ki

Secure Shell-SSH

három protokollból áll, amelyek a TCP felett futnak:



- szállítás protokoll (Transport Layer Protocol): a szerver hitelesítését, az adatok bizalmas kezelését és az adatok integritását biztosítja, adott esetben tömörítést is lehetővé tesz
- felhasználó hitelesítés protkoll (User Authentication Protocol): biztosítja a felhasználó hitelesítését a szerveren
- csatlakozás protokoll (Connection Protocol): több kommunikációs csatornát tud egyszerre üzemeltetni

SSH szállítás protokoll

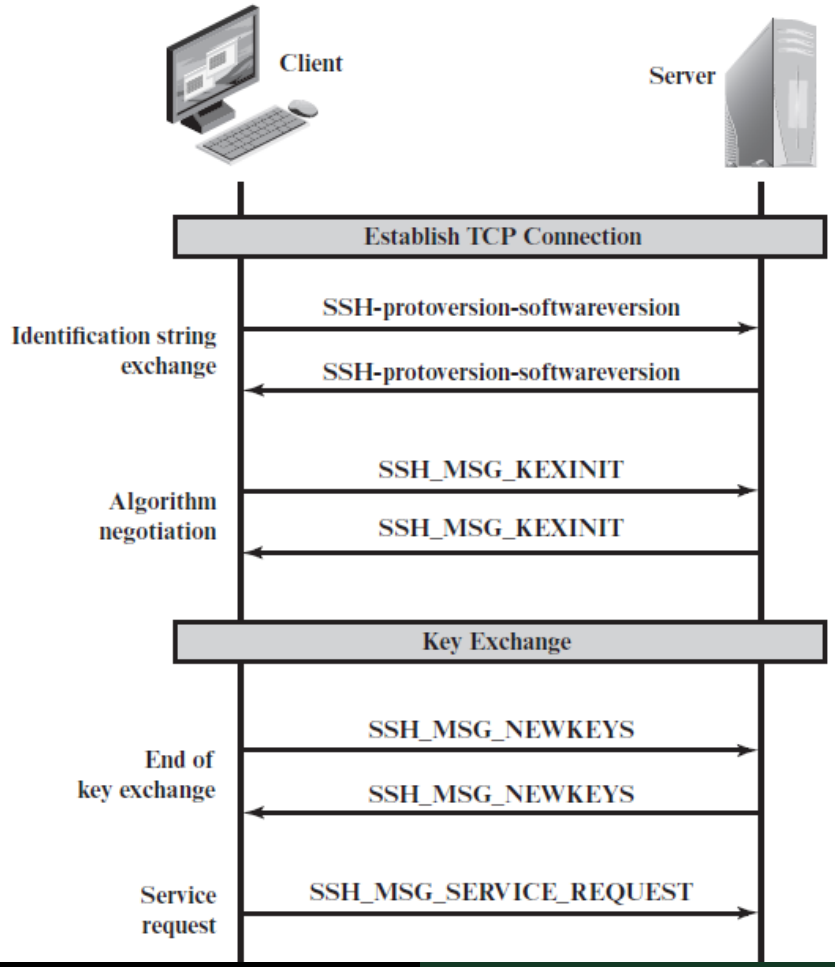
a szerver hitelesítésére a szállítási protokollban kerül sor:

- egy szerver (host-gép) több publikus/privát kulcspárral (host-kulcs) is rendelkezhet, mert többféle aszimmetrikus rendszert használhat
- több host-gép is megoszthatja ugyanazt a host-kulcsot.
- a kliensnek előzetesen ismernie kell a szerver publikus host-kulcsát, ezt kétféleképpen is meg lehet oldani:
  - a kliens egy lokális adatbázisban tárolja a host-gépneveket, a hozzájuk tarozó publikus host-kulcsokkal együtt. Nem igényel központi irányítást, hátrányos lehet az adatbázis karbantartása.
  - a host-gépnév és host-kulcsok társítását egy megbízható tanúsító hatóság (CA) hitelesíti. A kliens csak a CA gyökérkulcsát ismeri, így ellenőrizni tudja a host-gép kulcsának érvényességét. Könnyebb karbantartani, de az engedélyezés előtt minden host-kulcsot megfelelően hitelesítenie kell.

Az adat-integritás biztosításához forward secrecy-t használ: ha egy munkameneten belül egy kulcs értéke kompromitálódik, azaz nyilvánosságra kerül, az nem vonja maga után a korábbi titkosítások kompromitálását

SSH szállítás protokoll

A csomagcsere (packet exchange): a TCP protokollon keresztül a kliens kapcsolatot létesít a szerverrel, ezután kezdhető el a kliens és szerver közötti adat, azaz csomagcsere, amelynek lépései a következők:



SSH szállítás protokoll

A csomagcsere lépései:

- 1. azonosító karakterláncok cseréje (ID string exchange):
  - V_S a szerver, V_C a kliens azonosító karakterlánca
  - K_S a szerver publikus host-kulcsa
  - I_C a kliens, I_S pedig a szerver kezdeti üzenete
- 2. algoritmus egyeztetés (algorithm negotiation):
  - mindkét oldal elküldi a támogatott algoritmusok (kulcscsere, titkosítás, hash, MAC) listáját preferencia szerinti sorrendben
  - kiválasztásra az az algoritmus kerül, amely első helyen szerepel a kliens listáján, és amelyet a szerver is támogat
  - megállapodnak a q, α, z értékekben, ahol q egy nagy biztonságos prím, α a $GF(q)$ alcsoportjának generátora, z a $GF(q)$ rendje

SSH szállítás protokoll

Az csomagcsere lépései:

- 3. kulcscsere (key exchange):
  - a Diffie–Hellman kulcscserének két változatát lehet használni, leírásuk az RFC 2409 van meg
  - eredményként a két oldal osztozik egy K mesterkulcsban, amelynek a felhasználásával kerülnek meghatározásra a titkosító algoritmus titkos kulcsa, a MAC kulcs, illetve IV értékek
  - a H hash értéket munkamenet azonosítóként használják

SSH szállítás protokoll

Az csomagcsere lépései:

- 3. kulcscsere (key exchange):
  - C (a kliens) generál egy X_C random számot ($1 < X_C < z$), és kiszámítja az $Y_C = \alpha^{X_C} \pmod q$-t, amelyet elküld S-nek (szerver)
  - S generál egy X_S random számot ($1 < X_S < z$), és kiszámítja az $Y_S = \alpha^{X_S} \pmod q$-t, amelyet elküld C-nek
  - S kiszámítja :
    - $K = Y_C^{X_S} \pmod q$
    - $H = hash(V_C||V_S||I_C||I_S||K_S||Y_C||Y_S||K)$
    - meghatározza a host-privát kulcsával s-et, azaz H-ra a digitális aláírását
  - S elküldi a $(K_S||Y_S||s)$ értéket C-nek
  - C ellenőrzi vagy sem(!!), hogy a K_S valóban az S publikus host-kulcsa (az ellenőrzés hiánya sebezhetővé teszi a protokollt)
  - C kiszámítja:
    - $K = Y_S^{X_C} \pmod q$
    - $H = hash(V_C||V_S||I_C||I_S||K_S||Y_C||Y_S||K)$ értékeket
  - C ellenőrzi az s értékét

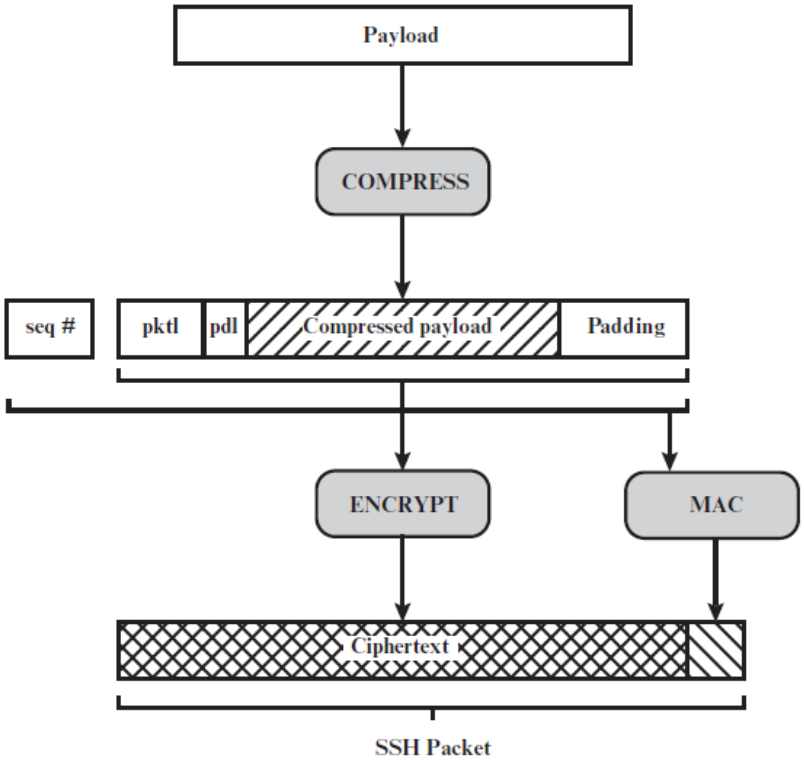
SSH szállítás protokoll

Az csomagcsere lépései:

- 4. kulcscsere vége (end of key exchange): mindkét fél jelzi, hogy lezárult a kulcscsere
- 5. szolgáltatási kérelem (service request): a kliens kéri a hitelesítési vagy csatlakozási protokoll elindítását

SSH szállítás protokoll

A csomagcserében minden csomag standard formátumú:



- pktl: a csomag hossza bájtban
- pdl: a random padding hossza
- Payload: a csomag hasznos tartalma
- Padding: a titkosítási algoritmus egyeztetése után kerül sor ennek a meghatározására, véletlenszerűen generált bájtokat tartalmaz
- MAC: üzenet hitelesítési kód, amelyet a rendszer a teljes csomagra és egy sorszámra (sequence number) számít ki, kivéve a MAC mezőt
- seq#: sorszám, egy implicit 32 bites érték, amely az első csomag esetében nulla, és minden csomag után egyel növekszik.

SSH szállítás protokoll

Az alkalmazható algoritmusok:

Cipher	
3des-cbc*	Three-key 3DES in CBC mode
blowfish-cbc	Blowfish in CBC mode
twofish256-cbc	Twofish in CBC mode with a 256-bit key
twofish192-cbc	Twofish with a 192-bit key
twofish128-cbc	Twofish with a 128-bit key
aes256-cbc	AES in CBC mode with a 256-bit key
aes192-cbc	AES with a 192-bit key
aes128-cbc**	AES with a 128-bit key
Serpent256-cbc	Serpent in CBC mode with a 256-bit key
Serpent192-cbc	Serpent with a 192-bit key
Serpent128-cbc	Serpent with a 128-bit key
arcfour	RC4 with a 128-bit key
cast128-cbc	CAST-128 in CBC mode

* = Required
** = Recommended

MAC algorithm	
hmac-sha1*	HMAC-SHA1; digest length = key length = 20
hmac-sha1-96**	First 96 bits of HMAC-SHA1; digest length = 12; key length = 20
hmac-md5	HMAC-MD5; digest length = key length = 16
hmac-md5-96	First 96 bits of HMAC-MD5; digest length = 12; key length = 16

Compression algorithm	
none*	No compression
zlib	Defined in RFC 1950 and RFC 1951

SSH felhasználó hitelesítés protokoll

- feladata a kliens hitelesítése, amelyhez több módszert is biztosít
- a hitelesítést a kliens irányítja, a szerver csupán a kliens hitelesítési kéréseire válaszol
- az üzenetváltás lépései:
  - a kliens küld egy `SSH_MSG_USERAUTH_REQUEST` üzenetet
  - a szerver ellenőrzi, hogy érvényes-e a felhasználónév, ha nem érvényes akkor egy `SSH_MSG_USERAUTH_FAILURE` üzenetet küld vissza
  - ha érvényes a felhasználónév, akkor a szerver elküldi a használható hitelesítési módszerek listáját
  - a kliens kiválasztja a hitelesítési módszert, és elküld egy `SSH_MSG_USERAUTH_REQUEST` üzenetet. Ezen a ponton előfordulhat, hogy további üzenetváltásra kerül sor
  - ha több hitelesítési módszerre is szükség van, akkor a szerver ezeket is ellenőrzi
  - amikor az összes szükséges hitelesítési módszer sikeres, a szerver egy `SSH_MSG_USERAUTH_SUCCESS` üzenetet küld, és ezzel a hitelesítési protokoll véget ér

SSH felhasználó hitelesítés protokoll

hitelesítési módszerek:

- egyszerű jelszó alapú hitelesítési módszerek:
  - lehetőség van a jelszó megváltoztatására
  - nincs minden programban implementálva
- publikus kulcsú kriptográfián alapuló módszerek: a támogatott rendszerek a DSA, ECDSA vagy RSA, illetve X.509 tanúsítványok
- billentyűzet-interaktív (RFC 4256) módszerek:
  - a szerver egy vagy több kérést kezdeményez, amelyeket a kliens megválaszol
  - one-time password típusú hitelesítés alkalmazását jelenti, például S/Key vagy SecurID
  - egyes OpenSSH-konfigurációk használják, megakadályozva az egyszerű jelszó-hitelesítési módszert
- GSSAPI (Generic Security Sevice Application Program Interface) hitelesítési módszerek:
  - külső alkalmazásokat vesznek igénybe, például Kerberos 5 vagy NTLM,
  - egyszeri bejelentkezési lehetőséget biztosítanak az SSH-munkamenetekhez
  - kereskedelmi SSH-megvalósításokban találjuk meg ezeket a megoldásokat

SSH csatlakozás protokoll

Alapfogalmak: csatorna, csatornakérések, globális kérések

- egyetlen SSH-kapcsolat egyszerre több csatornát is üzemeltethet, ahol mindegyik mindkét irányban továbbíthat adatot
- bármelyik oldal nyithat egy csatornát, ahol mindegyikhez, minden oldal egyedi csatornaszámot rendel
- a csatornák adat-szabályozása ablak mechanizmussal történik: addig nem lehet adatot küldeni egy csatornára, amíg nem érkezik egy olyan üzenet, amely azt jelzi, hogy van szabad ablak
- a csatornakérések a sávon kívüli csatorna-specifikus adatok továbbítására szolgálnak, például megváltozott terminálablak mérete, egy szerveroldali folyamat kilépési kódja. Minden csatorna saját vezérlést hajt végre.
- a globális kérés szerveroldali port kérését jelenti

SSH csatlakozás protokoll

Egy csatorna állapotának az esetében három szakaszt különböztetünk meg: a csatorna megnyitása, adatátvitel és a csatorna bezárása.

Megnyítás: az `SSH_MSG_CHANNEL_OPEN` üzenettel történik és a következőket kell specifikálni:

- a csatorna típusát, számát
- a kezdeti ablakméretet: megadja, hogy hány bájtnyi adatot lehet küldeni a feladónak
- a maximum csomagméretet: megadja a feladónak elküldhető egyedi adatcsomag maximális méretét. Például érdemes lehet kisebb csomagokat használni az interaktív kapcsolatokhoz.

Ha a távoli oldal meg tudja nyitni a csatornát, akkor egy `SSH_MSG_CHANNEL_OPEN_CONFIRMATION` üzenetet küld:

- csatorna szám
- a fogadó csatorna szám
- a bejövő forgalom ablak- és csomagméretét

Ha nem tudja megnyitni a csatornát hibaüzenet küld.

SSH csatlakozás protokoll

Adatátvitel:

- az `SSH_MSG_CHANNEL_DATA` üzenettel történik, amely tartalmazza a fogadó csatorna számát és egy adatblokkot.
- az üzenetek mindkét irányban mindaddig fennmaradhatnak, amíg a csatorna nyitva van.

Zárás:

- Ha valamelyik fél be akar zárni egy csatornát, `SSH_MSG_CHANNEL_CLOSE` üzenetet küld, amely tartalmazza a címzett csatorna számát.

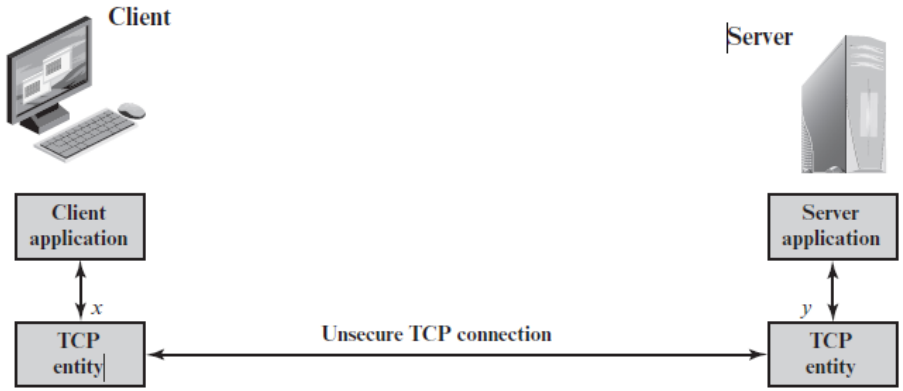
SSH csatlakozás protokoll

Port forwarding:

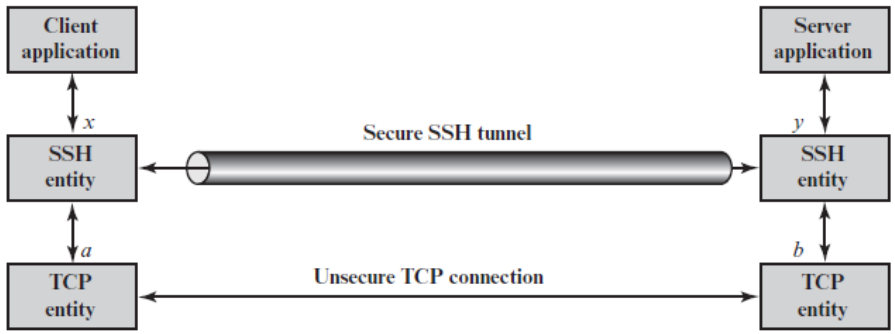
- az egyik leghasznosabb SSH funkció
- bármilyen nem biztonságos TCP-kapcsolat biztonságos SSH-kapcsolattá alakítható: SSH-tunneling (SSH-alagút)
- port: a TCP felhasználó azonosítója, minden TCP-n futó alkalmazásnak van egy portszáma.
- a bejövő TCP-forgalom a portszám alapján kerül a megfelelő alkalmazáshoz. Például a Simple Mail Transfer Protocol (SMTP) esetében a szerver általában a 25-ös portot figyeli, és az adatokat a 25-ös portra küldi. A TCP felismeri, hogy ez az SMTP-szerver címe, és az adatokat az SMTP-szerver alkalmazáshoz irányítja.
- egy alkalmazás több portszámot is használhat
- kétféle porttovábbítás: helyi továbbítás és távoli továbbítás

SSH csatlakozás protokoll

A Port forwarding alapkoncepciója:



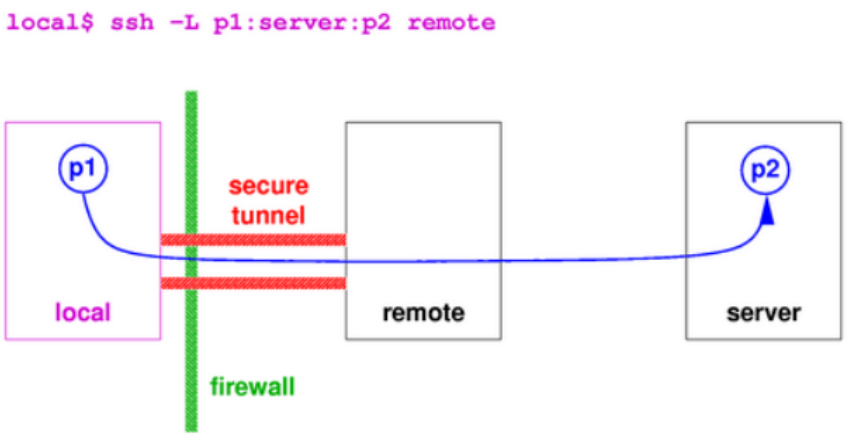
(a) Connection via TCP



SSH csatlakozás protokoll

Helyi port továbbítás (local forwarding)

- a kliens beállíthat egy „hijacker” folyamatot: a kiválasztott alkalmazásszintű forgalmat, átirányítja egy nem biztonságos TCP-kapcsolatról egy biztonságos SSH-alagútba
- a felhasználó biztonságosan csatlakozhat a helyi számítógépéről egy távoli gépre



SSH csatlakozás protokoll

Helyi port továbbítás (local forwarding)

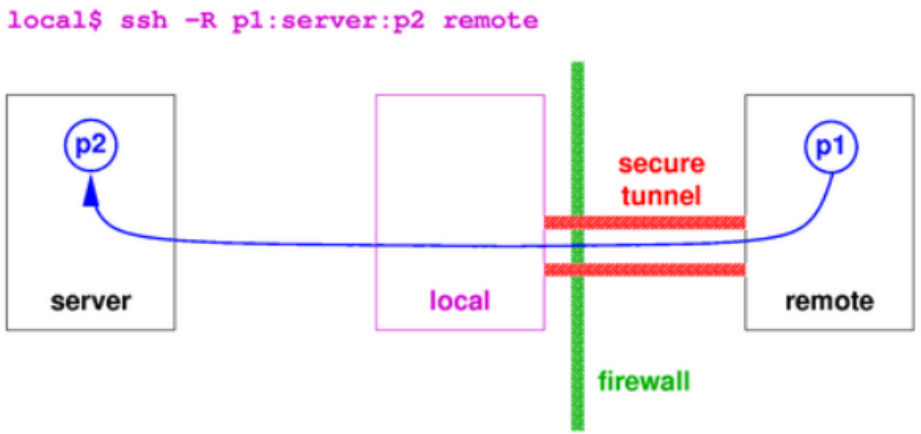
Példa: a Post Office Protocol (POP3) levelező szervert használva biztonságos levelezést akarunk beállítani SSH protokollt használva. A POP3-hoz hozzárendelt portszám a 110-es port. A beállítás lépéssorozata a következő lesz:

- az SSH-kliens kiválaszt egy nem használt helyi portszámot, például 9999-et, és beállítja az SSH-t úgy, hogy az adatforgalmat továbbítsa egy SSH alagúton keresztül először az SSH-szerverre, onnan pedig a 110-es porton keresztül a távoli levelezőszerverre
- az adatok tehát először a 9999-re mennek onnan pedig titkosítva a SSH szerverre, amely továbbítja őket a 110-es porton keresztül a távoli levelezőszerverre
- a forwarded port a 9999 lesz

SSH csatlakozás protokoll

Távoli port továbbítás (remote forwarding):

- lehetővé teszi, hogy az SSH kapcsolat szerver oldalán lévő alkalmazások hozzáférjenek az SSH kliens oldalán található szolgáltatásokhoz.
- más szóval: lehetővé teszi a felhasználók számára, hogy egy SSH alagút szerveroldaláról csatlakozzanak egy távoli hálózati szolgáltatáshoz



SSH csatlakozás protokoll

Távoli port továbbítás (remote forwarding):

Példa: egy felhasználó egy munkahelyi szervert szeretne elérni otthoni számítógépéről. A munkahelyi szerver tűzfal mögött van, ezért nem fogad SSH-kérést az otthoni számítógépről. A munkahelyről azonban beállíthat egy SSH-alagutat:

- a munkahelyi számítógépről beállít egy SSH-kapcsolatot az otthoni számítógépéhez. A tűzfal ezt lehetővé teszi, mert ez egy védett kimenő kapcsolat
- beállítja az SSH-szervert úgy, hogy az egy helyi portot figyeljen, mondjuk a 22-es portot, és adatokat továbbítson az SSH-kapcsolaton keresztül a távoli porthoz, például a 2222-es porthoz
- az otthoni számítógépén úgy állítja be az SSH-t hogy az a 2222-es porton fogadja az adatokat
- a létrehozott SSH-alagúton, a munkahelyi szerverre távoli bejelentkezést alkalmazhat