Securitatea Sistemelor Informatice

- Curs 13 -
TLS

Adela Georgescu

Facultatea de Matematică și Informatică
Universitatea din București
Anul universitar 2022-2023, semestrul I

TLS - Transport Layer Security

- Este un protocol folosit de browser-ul web de fiecare data cand ne conectam la un browser folosind `https`

TLS - Transport Layer Security

- Este un protocol folosit de browser-ul web de fiecare data cand ne conectam la un browser folosind `https`
- primele versiuni se numeau SSL - Secure Sockets Layer - dezvoltat de Netscape (1995) - SSL 3.0, cea mai cunoscută versiune
- TLS 1.0 apare în 1999, TLS 1.1 în 2006, TLS 1.2 în 2008 și versiunea actuală, sigura și eficientă TLS 1.3 în 2018

TLS - Transport Layer Security

- Este un protocol folosit de browser-ul web de fiecare data cand ne conectam la un browser folosind `https`
- primele versiuni se numeau SSL - Secure Sockets Layer - dezvoltat de Netscape (1995) - SSL 3.0, cea mai cunoscută versiune
- TLS 1.0 apare în 1999, TLS 1.1 în 2006, TLS 1.2 în 2008 și versiunea actuală, sigura și eficientă TLS 1.3 în 2018
- folosirea lui SSL 3.0, TLS 1.0 și TLS 1.1 trebuie evitată, toate trei prezintă probleme de securitate
- se recomandă folosirea minim a lui TLS 1.2

TLS - Transport Layer Security

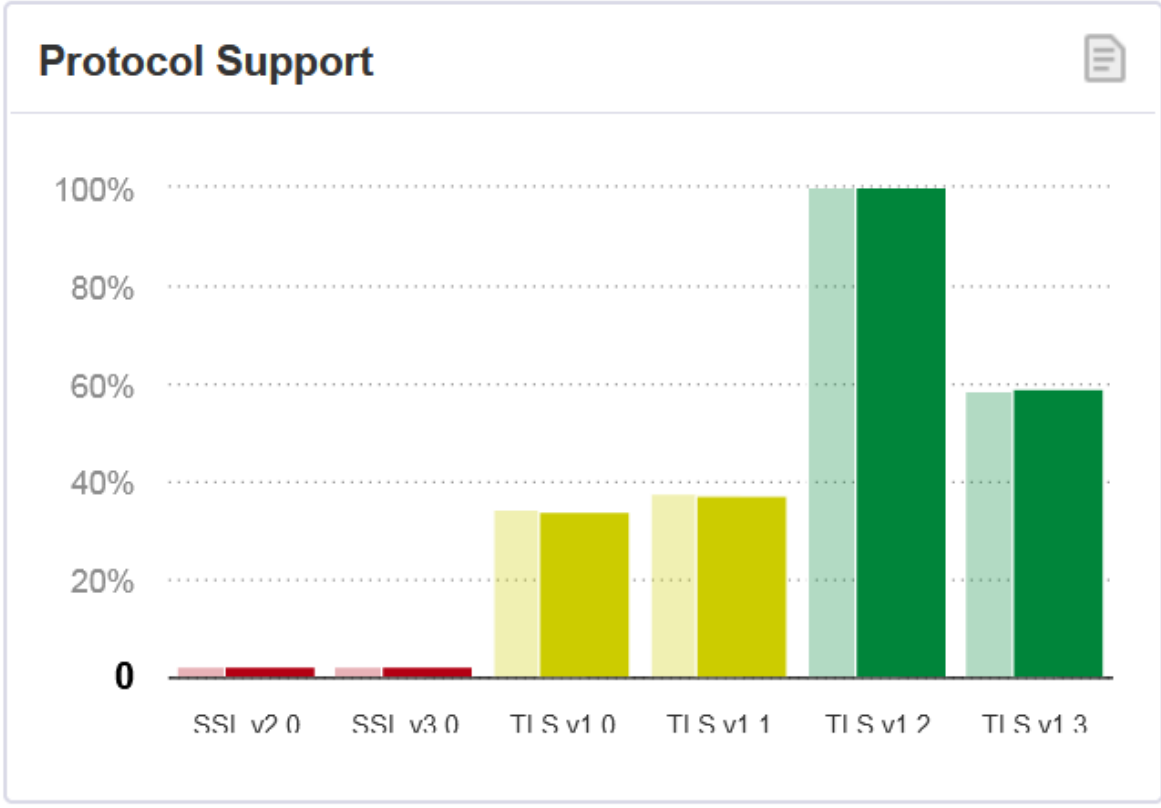
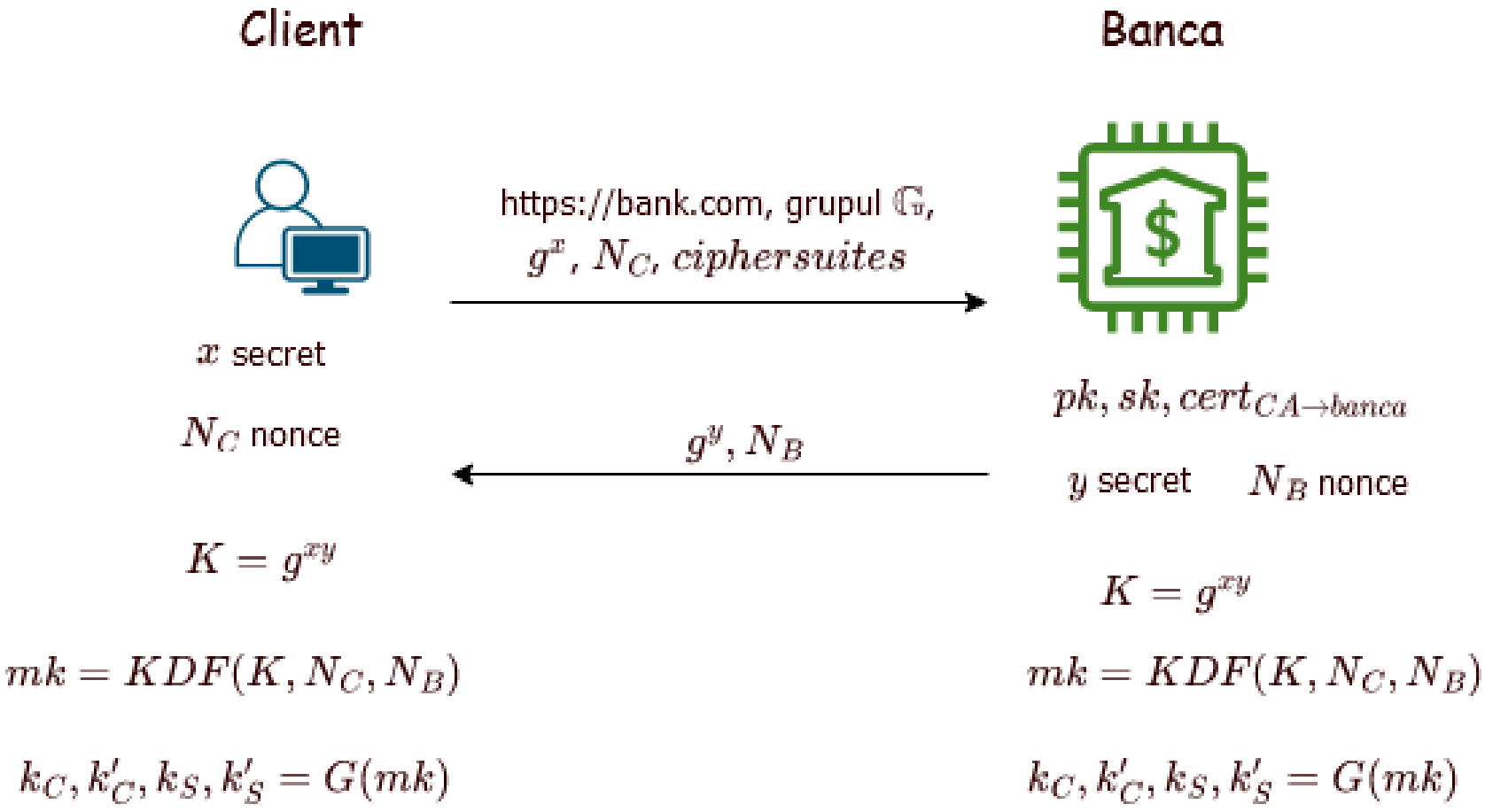


Figure: Acoperirea versiunilor de TLS conform SSL Pulse (decembrie 2022)

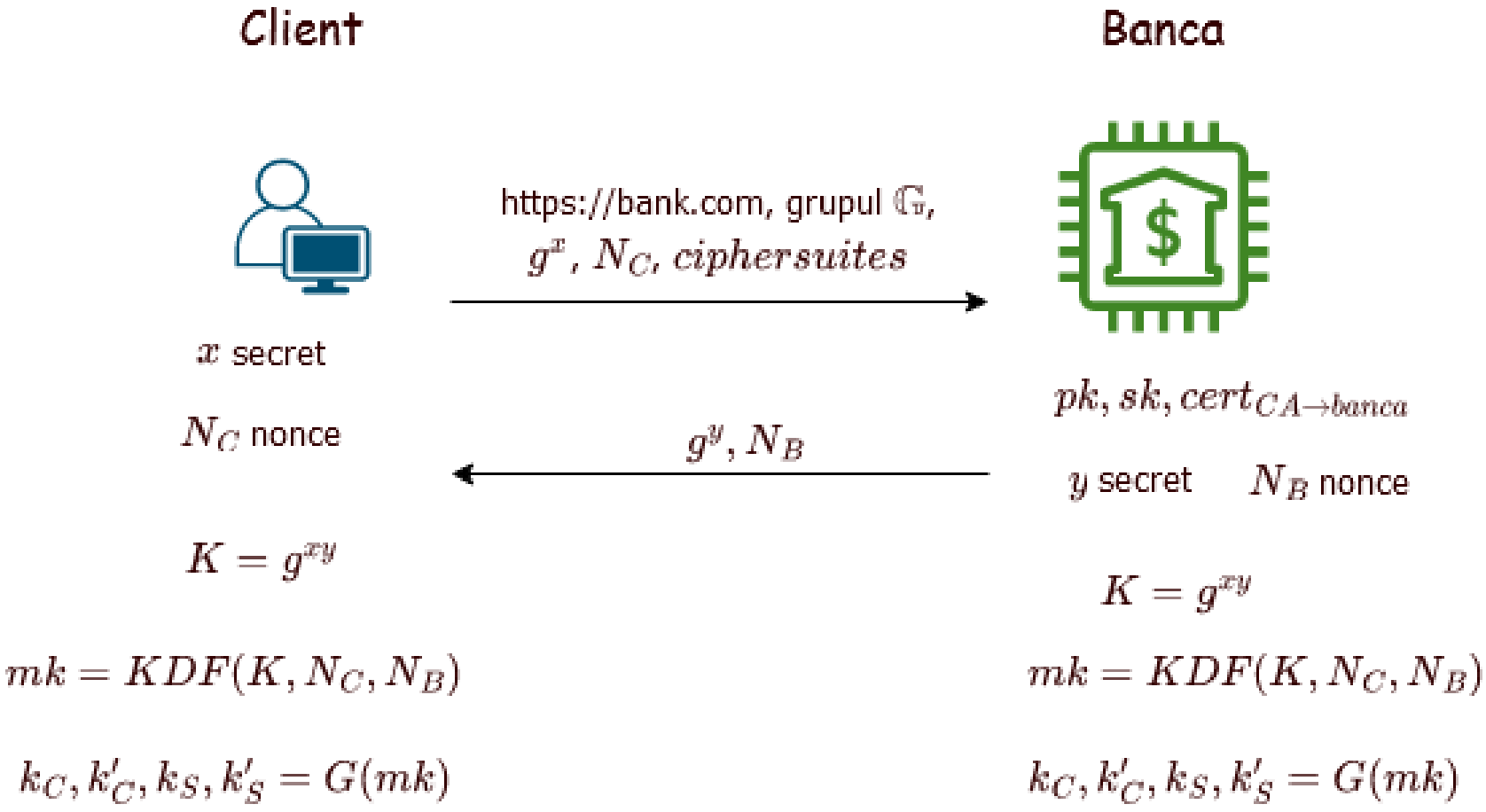
TLS - Transport Layer Security

- Protocolul TLS permite unui client (de ex. browser web) și unui server (de ex. website) să se pună de acord asupra unui set de chei pe care să le folosească ulterior pentru comunicare criptată si autentificare
- Protocolul TLS constă din 2 părți:
  1. *protocolul handshake* - realizează schimbul de chei care stabilește un set de chei comune
  2. *protocolul record-layer* - folosește cheile stabilite pentru criptare/autentificare ulterioară
- In continuare vom prezenta protocolul handshake din versiunea curentă TLS 1.3

Handshake protocol (TLS 1.3)...

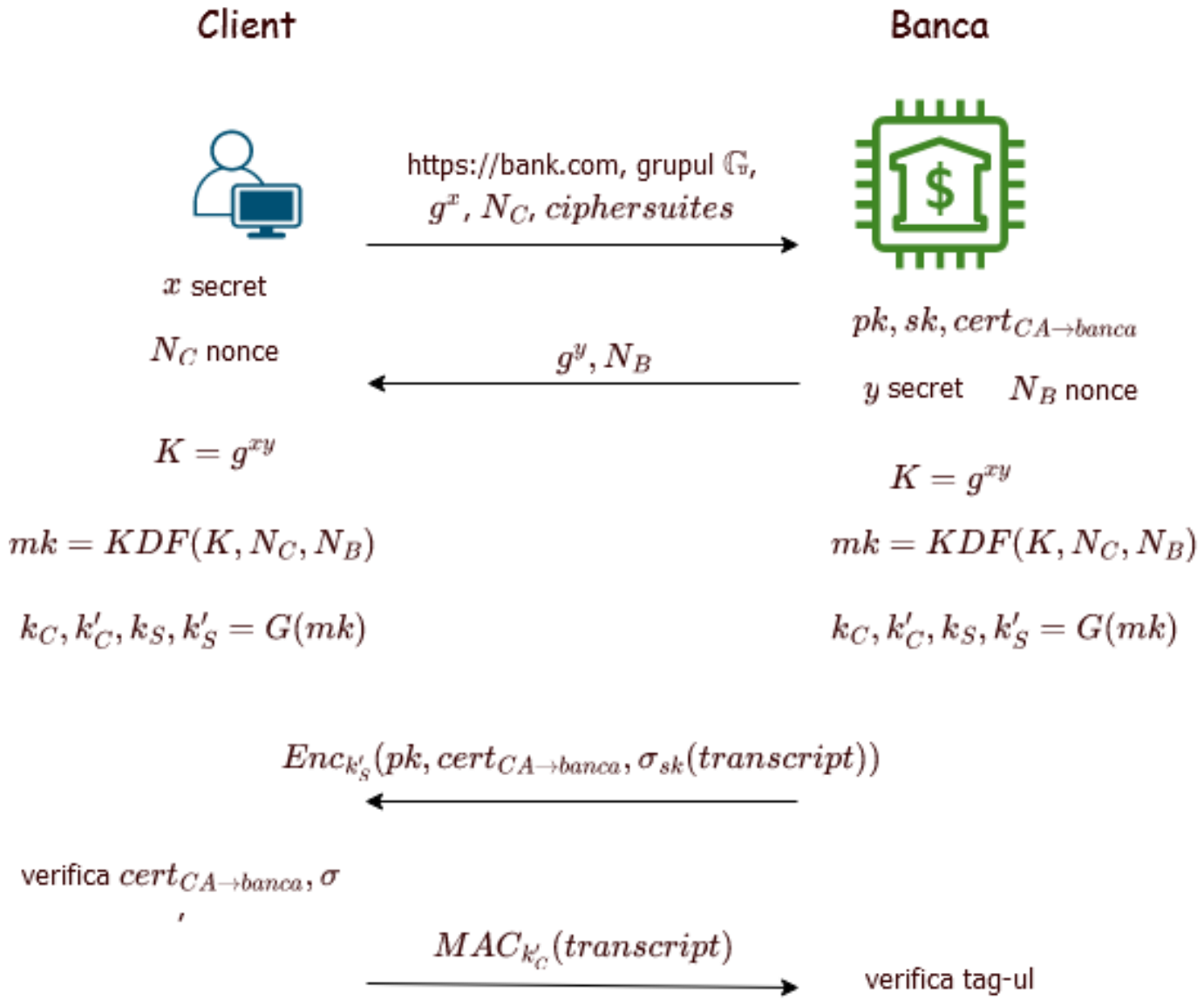


Handshake protocol (TLS 1.3)...

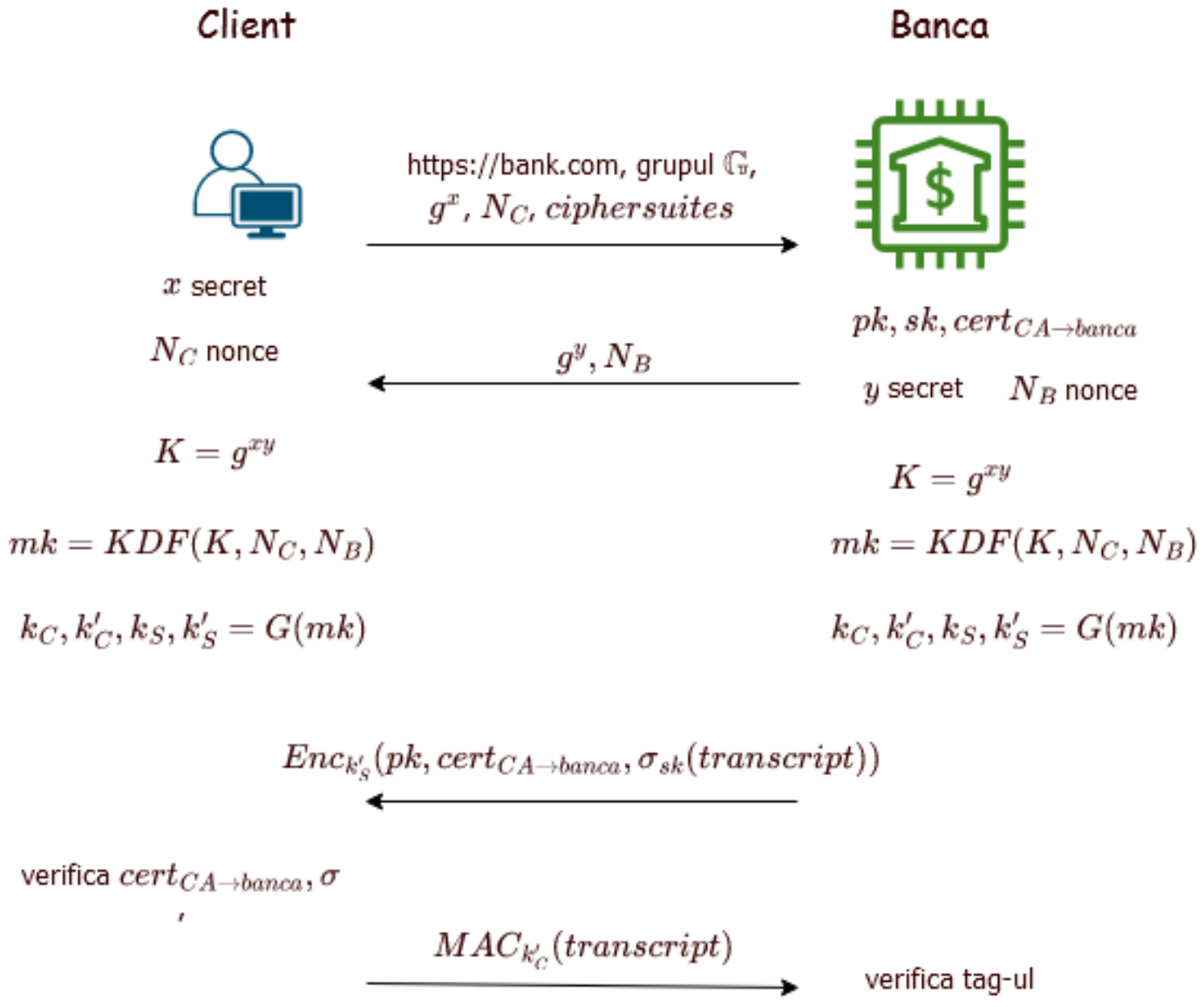


- KDF = key derivation function - algoritm criptografic pentru derivarea de chei, pe baza unui PRF
- G = generator de numere pseudo-aleatoare (PRG)

...Handshake protocol...



...Handshake protocol...



...Handshake protocol

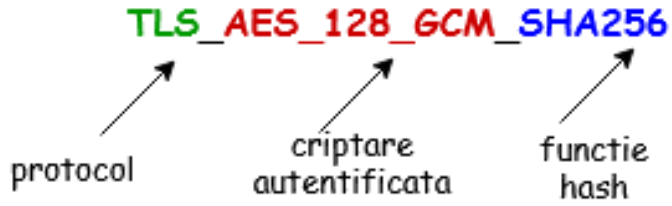
- *ciphersuites* - colecție de algoritmi criptografici

...Handshake protocol

- *ciphersuites* - colecție de algoritmi criptografici
- TLS 1.3 suportă 5 ciphersuites:

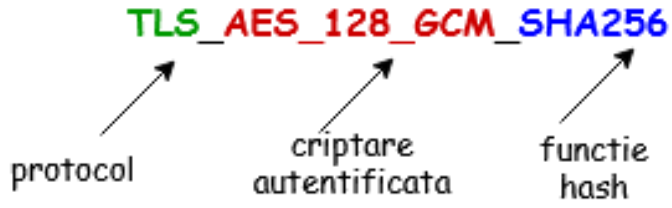
...Handshake protocol

- *ciphersuites* - colecție de algoritmi criptografici
- TLS 1.3 suportă 5 ciphersuites:
  - TLS_AES_128_GCM_SHA256
  - TLS_AES_256_GCM_SHA384
  - TLS_CHACHA20_POLY1305_SHA256
  - TLS_AES_128_CCM_SHA256
  - TLS_AES_128_CCM_8_SHA256



...Handshake protocol

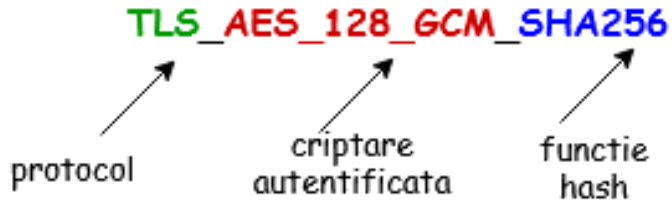
- *ciphersuites* - colecție de algoritmi criptografici
- TLS 1.3 suportă 5 ciphersuites:
  - TLS_AES_128_GCM_SHA256
  - TLS_AES_256_GCM_SHA384
  - TLS_CHACHA20_POLY1305_SHA256
  - TLS_AES_128_CCM_SHA256
  - TLS_AES_128_CCM_8_SHA256



- *transcript* - reprezintă mesajele trimise în cadrul protocolului până la momentul curent

...Handshake protocol

- *ciphersuites* - colecție de algoritmi criptografici
- TLS 1.3 suportă 5 ciphersuites:
  - TLS_AES_128_GCM_SHA256
  - TLS_AES_256_GCM_SHA384
  - TLS_CHACHA20_POLY1305_SHA256
  - TLS_AES_128_CCM_SHA256
  - TLS_AES_128_CCM_8_SHA256



- *transcript* - reprezintă mesajele trimise în cadrul protocolului până la momentul curent
- cheile k'_S și k'_C sunt folosite doar în handshake

TLS 1.3

- clientul și serverul, partajează, la finalul protocolului, cheile k_S și k_C, pe care le folosesc ulterior pentru criptare și autentificare

TLS 1.3

- clientul și serverul, partajează, la finalul protocolului, cheile k_S și k_C, pe care le folosesc ulterior pentru criptare și autentificare
- în plus, clientul are garanția că la final a partajat cheile cu server-ul legitim și că acestea nu au fost interceptate sau modificate de o terță parte

TLS 1.3

- clientul și serverul, partajează, la finalul protocolului, cheile k_S și k_C, pe care le folosesc ulterior pentru criptare și autentificare
- în plus, clientul are garanția că la final a partajat cheile cu server-ul legitim și că acestea nu au fost interceptate sau modificate de o terță parte de ce?

TLS 1.3

- clientul și serverul, partajează, la finalul protocolului, cheile k_S și k_C, pe care le folosesc ulterior pentru criptare și autentificare
- în plus, clientul are garanția că la final a partajat cheile cu server-ul legitim și că acestea nu au fost interceptate sau modificate de o terță parte de ce?
- în handshake-ul din TLS 1.2, clientul și server-ul foloseau o schemă de criptare cu cheie publică în locul schimbului de chei Diffie-Hellman

TLS 1.3

- clientul și serverul, partajează, la finalul protocolului, cheile k_S și k_C, pe care le folosesc ulterior pentru criptare și autentificare
- în plus, clientul are garanția că la final a partajat cheile cu server-ul legitim și că acestea nu au fost interceptate sau modificate de o terță parte de ce?
- în handshake-ul din TLS 1.2, clientul și server-ul foloseau o schemă de criptare cu cheie publică în locul schimbului de chei Diffie-Hellman
- clientul doar alegea o cheie K pe care o trimitea serverului criptată cu cheia lui publică

TLS 1.3

- aceasta a fost eliminată în TLS 1.3 pentru că nu asigură proprietatea de *forward secrecy* - care presupune că, în cazul compromiterii server-ului, cheile de sesiune anterioare nu sunt compromise

TLS 1.3

- aceasta a fost eliminată în TLS 1.3 pentru că nu asigură proprietatea de *forward secrecy* - care presupune că, în cazul compromiterii server-ului, cheile de sesiune anterioare nu sunt compromise

- în TLS 1.3, în ceea ce priveste server-ul, cheia K este calculata pe baza secretului y in fiecare sesiune, secret care este unul nou de fiecare data si care nu trebuie mentinut intre sesiuni; deci compomiterea server-ului (aflarea cheii lui secrete) nu duce la aflarea cheii K

TLS 1.3

- aceasta a fost eliminată în TLS 1.3 pentru că nu asigură proprietatea de *forward secrecy* - care presupune că, în cazul compromiterii server-ului, cheile de sesiune anterioare nu sunt compromise

- în TLS 1.3, în ceea ce priveste server-ul, cheia K este calculata pe baza secretului y in fiecare sesiune, secret care este unul nou de fiecare data si care nu trebuie mentinut intre sesiuni; deci compomiterea server-ului (aflarea cheii lui secrete) nu duce la aflarea cheii K

- în TLS 1.2, cheia secretă K ii este trimisa server-ului criptata cu cheia lui publica; compromiterea cheii secrete server-ului duce la compromiterea cheilor de sesiune din toate sesiunile

TLS 1.3

- aceasta a fost eliminată în TLS 1.3 pentru că nu asigură proprietatea de *forward secrecy* - care presupune că, în cazul compromiterii server-ului, cheile de sesiune anterioare nu sunt compromise

- în TLS 1.3, în ceea ce priveste server-ul, cheia K este calculata pe baza secretului y in fiecare sesiune, secret care este unul nou de fiecare data si care nu trebuie mentinut intre sesiuni; deci compomiterea server-ului (aflarea cheii lui secrete) nu duce la aflarea cheii K

- în TLS 1.2, cheia secretă K ii este trimisa server-ului criptata cu cheia lui publica; compromiterea cheii secrete server-ului duce la compromiterea cheilor de sesiune din toate sesiunile

- în protocolul *record*, clientul și server-ul comunică în mod confidențial și autentificat folosind o schemă de criptare autentificată

TLS 1.3 vs. TLS 1.2

- număr redus de runde

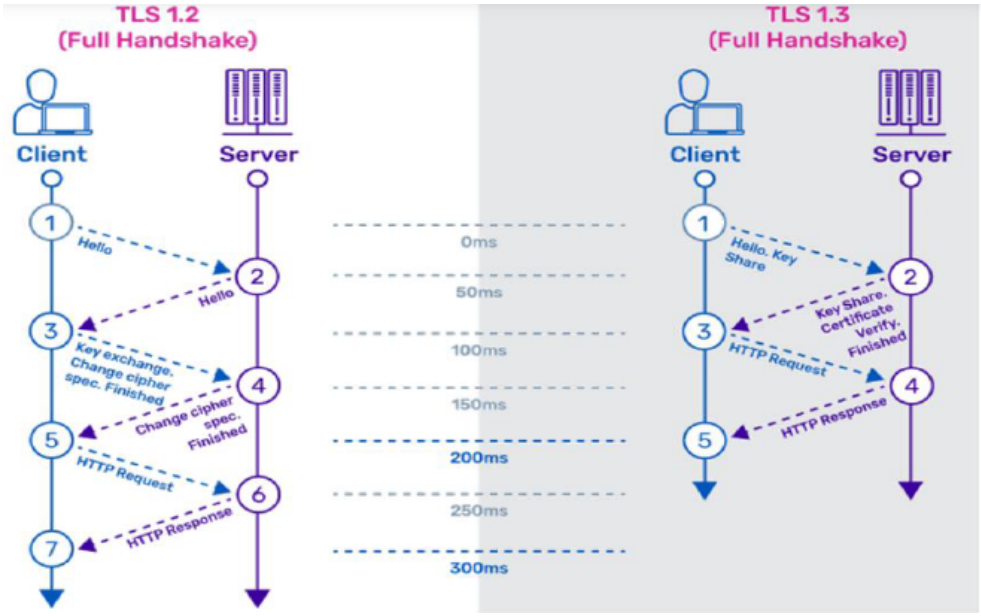


Figure: Sursa: www.a10networks.com

TLS 1.3 vs. TLS 1.2

- număr redus de runde

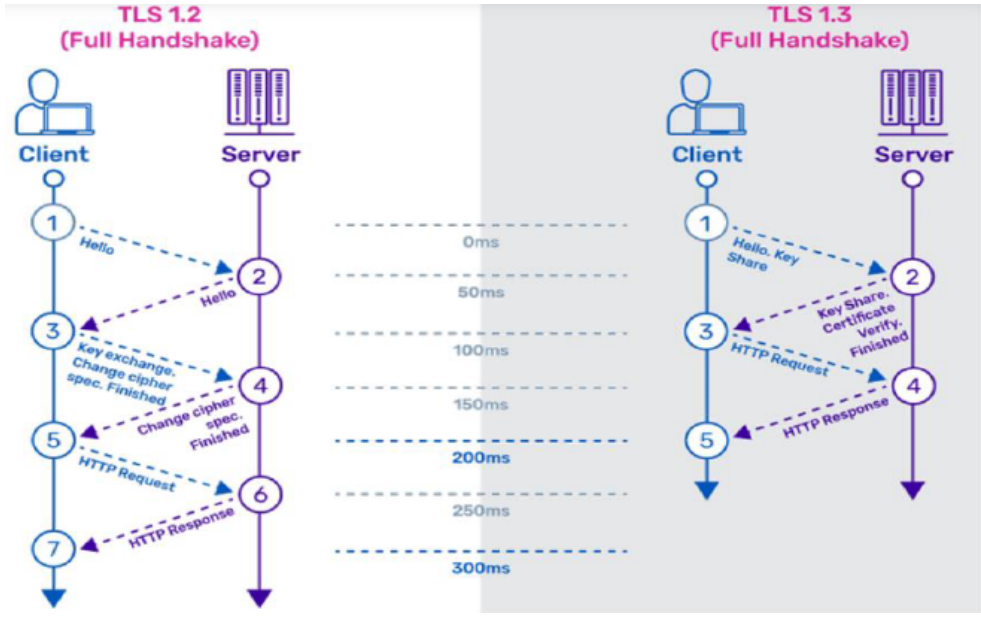


Figure: Sursa: www.a10networks.com

- eliminarea algoritmilor criptografici vulnerabili precum SHA-1, RC4, DES, 3DES, AES-CBC, MD5

TLS 1.3 vs. TLS 1.2

- număr redus de runde

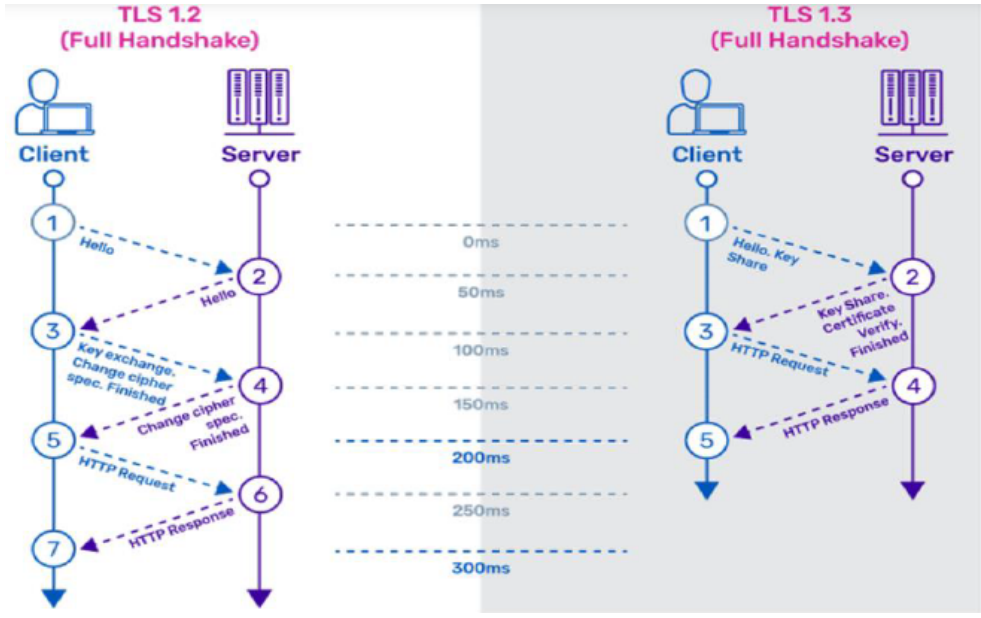


Figure: Sursa: www.a10networks.com

- eliminarea algoritmilor criptografici vulnerabili precum SHA-1, RC4, DES, 3DES, AES-CBC, MD5
- 0-RTT (zero round-trip) – reluarea unei sesiuni mai vechi cu un website e mult mai rapida