Needham-Schroederův protokol

* základní varianta Needham a Schroeder 1978
* zajímavý zejména z historických důvodů, protože je základem mnoha autentizačních protokolů a protokolů pro distribuci klíčů, jeho praktické použití se ale nedoporučuje (ukážeme nedostatek)

1. Alice chce komunikovat s Bobem, pošle zprávu KDC obsahující nonci r_A: (r_A, A, B)
2. KDC pošle zpátky zprávu obsahující nonci Alice, relační klíč K a ticket který Alice může zaslat Bobovi: K_A(r_A, B, K, K_B(A, K))
   - Alice zkontroluje nonci r_A - ví zda její zpráva nebo replay
   - Alice zkontroluje B - ví, že Oskar zprávu (1) nezachytil a nenahradil v ní B vlastní identitou (aby KDC vyrobilo ticket pro Oskara)
3. Alice pošle ticket Bobovi spolu s noncí r_A2 zašifrovaným relačním klíčem K: (K_B(A, K), K(r_A2))
4. Bob pošle zpět (K(r_A2 - 1), r_B); protože Alice sama vygenerovala čerstvé r_A2, příchodem K(r_A2 - 1) ověří že mluví s opravdovým Bobem a není to replay
5. Alice pošle zpět K(r_B - 1), tím Bob ověří že mluví s Alicí a není to replay

* protokol má slabost: pokud Oskar zjistí některý starý relační klíč, může iniciovat novou relaci s Bobem přehráním zprávy (3) (Denningová a Sacco 1981)

* Needham a Schroeder opravili v 1987, ve stejném čísle časopisu ”OS Review” publikován trochu jednodušší Otway-Reesův protokol, který si uvedeme

Otway-Reesův protokol

* Ottway a Rees 1987

1. Alice vygeneruje dvojici náhodných noncí r (společný identifikátor transakce) a r_A (výzva Bobovi)
   - pošle Bobovi zprávu: (r, A, B, K_A(r_A, r, A, B))
2. Bob obdrží zprávu, z přijatých dat a nové nonce r_B vytvoří analogickou zprávu K_B(r_B, r, A, B)
   - obě pošle KDC: (r, A, B, K_A(r_A, r, A, B), K_B(r_B, r, A, B))
3. KDC zkontroluje A a B a zda je r v obou zprávách stejné
   - pokud ne, asi se Oskar snaží o replay nebo změnil část zprávy (1) nebo (2)
   - pokud ano, KDC věří Bobovu požadavku; vygeneruje relační klíč K a zašifruje ho jednou pro Alici a jednou pro Boba
   - KDC pošle Bobovi zprávu: (K_A(r_A, K), K_B(r_B, K))
4. Bob dešifruje druhou část zprávy, ověří že r_B odpovídá zaslanému v (2)
   - pokud ano, Bob pošle Alici zprávu: K_A(r_A, K)
   - Alice dešifruje, zkontroluje zda r_A odpovídá zaslanému v (1)
   - Alice i Bob vědí, že K je čerstvý a že druhá strana sdílí příslušný klíč s KDC; mohou spolu začít komunikovat

Kerberos

* používá se v mnoha reálných systémech, mimo jiné v Orionu
* Kerberos verze 1 až 4 (od 1987, MIT) přidáním časových razítek (podle doporučení Denningové a Sacca) do Needham-Schroederova protokolu
  - redukován počet zasílaných zpráv za cenu existence bezpečných synchronizovaných hodin
  - účelem dovolit uživatelům bezpečně přistupovat k síťovým zdrojům
  - Kerberos verze 5 v 1989 na základě kritiky verze 4
* uvedeme pouze kryptografické aspekty protokolu

* protokolu Kerberos se účastní:
  - Alice - proces uživatele na PC připojeném na síť
  - Bob - server provádějící službu požadovanou Alicí
  - Kerberos coby důvěryhodná třetí strana

. autentizační server Kerbera (authentication server, AS) - ověřuje uživatele při přihlašování
. ticket-granting server (TGS) - vydává tickety prokazující identitu

* Alice se přihlašuje do systému:
- Alice má společný klíč s Kerberem, klíč je odvozen z jejího hesla příslušnou jednosměrnou funkcí

1. Alice zadá své uživatelské jméno, jméno se odešle AS Kerbera
2. AS vygeneruje náhodný relační klíč K_AK a tzv. ticket-granting ticket; klíč K_AK a ticket-granting ticket jsou zašifrovány klíčem, odvozeným z uživatelského hesla (zná ho tedy pouze Alice a Kerberos) a zaslány Alici:

   K_passwd(K_AK, K_T(A, t_now, K_AK))

   (TGT obsahuje informaci o identitě A, časové razítko t_now a právě vytvořený relační klíč K_AK; ticket-granting ticket je zašifrován klíčem K_T, který zná pouze Kerberos)
3. Alice (resp. systém, do kterého se přihlásila) dešifruje odpověď a zruší klíč odvozený z hesla

* pokud požaduje Alice službu od některého serveru (Boba), musí nejprve získat od TGS ticket pro tuto službu
- po přihlášení Alice a Kerberos sdílejí relační klíč K_AK, Alice má ticket-granting ticket TGT
- Bob a Kerberos sdílejí klíč K_BK
- Alice autentizuje vůči Bobovi následujícím protokolem:

1. Alice vygeneruje nonci N_A a pošle AS Kerbera zprávu:

        (A, B, N_A)

   kde A je identifikátor Alice, B je identifikátor Boba, nonce N_A je náhodné číslo, vygenerované Alicí; nonce smí být použito právě jednou.

2. Kerberos vygeneruje nový relační klíč K, určí dobu platnosti L ticketu, a vytvoří ticket pro Boba: ticket_B = K_BK(K, A, L)
   Kerberos pošle Alici zprávu:

        (ticket_B, K_AK(K, N_A, L, B))

3. Alice dešifruje svou část zprávy klíčem K_AK a ověří, že identifikátor B a nonce N_A odpovídají položkám, zaslaným v (1). Alice uchová L a K. Pak vytvoří autentizátor:

        auth_A = K(A, T_A, Ks)

   kde T_A je čerstvé časové razítko a Ks je nepovinný tajný podklíč (jeho výhodou je, že nebyl vytvořen Kerberem). Alice pak pošle Bobovi zprávu:

        (ticket_B, auth_A)

4. Server Bob přijme zprávu (3), dešifruje ticket klíčem K_BK a použije K pro dešifrování autentizátoru. Pak ověří, že si odpovídají identifikátory v ticketu a autentizátoru, platnost časového razítka T_A v autentizátoru a porovná lokální čas s dobou platnosti L ticketu.

   Pokud jsou všechny testy úspěšné, je Alice autentizována vůči Bobovi.

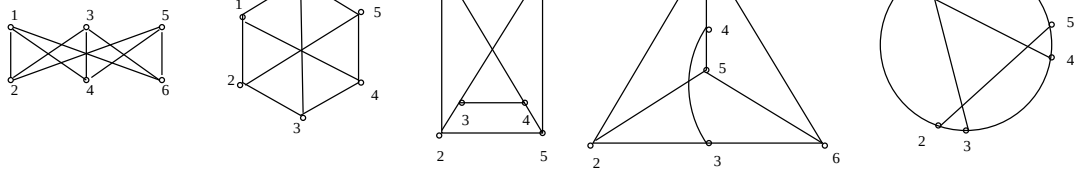
5. Pokud se Alice chce znovu autentizovat vůči Bobovi po dobu platnosti ticketu znovu, vytvoří nový autentizátor s čerstvým časovým razítkem; v takovém případě může být výhodné definovat nový tajný podklíč Ks.

* výše uvedené je základní autentizace zjednodušeného protokolu Kerberos V5 (některá pole protokolu jsou vynechána)
- základním cílem protokolu je ověřit identitu Alice
- vedlejším efektem protokolu je ustanovení klíče sdíleného Alicí a Bobem
- protokol předpokládá existenci bezpečných synchronizovaných hodin
- protokol závisí na bezpečnosti hesla

Protokoly s nulovou znalostí

* dosavadní identifikační protokoly vyžadovaly, aby Alice sdílela tajný klíč
* nevýhoda - druhá strana tajný klíč např. předat někomu dalšímu, ten pak může předstírat, že je Alice
* proto navrženy tzv. protokoly s nulovou znalostí, ang. zero-knowledge (ZK) identification protocols
  - nevyžadují šifrování, sekvenční čísla ani časová razítka
  - Alice demonstruje znalost nějakého tajemství, aniž by ho ověřovatel mohl zjistit a předat dalším

* jako příklad uvedu ZK protokol založený na obtížnosti hledání Hamiltonovských kružnic v grafu (Blum 1986)
  - Hamiltonovská kružnice je kružnice která prochází každým vrcholem grafu (kružnice = neorientovaná uzavřená cesta)
  - hledání Hamiltonovských kružnic v grafu je obtížný problém
  - rozhodnutí zda jsou dva grafy izomorfní je obtížný problém; např. jsou následující grafy izomorfní?



* Alice vytvoří graf G s Hamiltonovskou kružnicí, graf předá Bobovi
* Bob zná G, nezná ale Hamiltonovskou kružnici v G
* Alice se chce autentizovat znalostí Hamiltonovské kružnice, aniž by jí prozradila
* ukážu poněkud zjednodušenou verzi:

1. Alice vytvoří graf H izomorfní k G náhodnou permutací, graf H pošle Bobovi
2. Bob požádá Alici o jednu ze dvou možností:
   - buď požádá o důkaz že H je izomorfní k G
   - nebo požádá o odhalení Hamiltonovské kružnice v H
3. Alice pošle požadovanou odpověď
4. opakuje se, po N iteracích je pravděpodobnost že Alice podvádí (1/2)^N

Protokoly založené na asymetrických technikách

Diffie-Hellmanův protokol

* zatím jsme předpokládali, že Alice a Bob sdílejí tajný klíč
* pokud nesdílejí, jak se na něm dohodnou?

* tzv. protokoly pro dohadování klíče, umožňují ustanovit tajný klíč po otevřeném kanálu

Diffie-Hellmanův protokol (1976) je nejstarší a nejznámější prakticky používaný protokol, který dovoluje dvěma stranám vytvořit sdílený tajný klíč komunikací po otevřeném kanálu (síti). Účastníci komunikace nemusejí před započetím protokolu sdílet žádnou informaci. (Merkleho metoda kryptogramů je sice starší, ale pro praktické použití je výpočetně i datově příliš náročná.)

Základní verze Diffie-Hellmanova protokolu dohadování klíčů sestává z následujících kroků:

---

1. Alice a Bob se domluví na dvou velkých prvočíslech p a α, kde $(p-1)/2$ je také prvočíslo, $2 \leq \alpha \leq p-2$ a α musí splňovat ještě další podmínky. Čísla p a α nemusí být tajná, tj. Alice nebo Bob je mohou zvolit a poslat druhé straně otevřeným kanálem.
2. Alice zvolí náhodné tajné číslo x, $1 \leq x \leq p-2$. Alice pošle Bobovi zprávu, obsahující hodnotu $\alpha^x \bmod p$.

3. Bob zvolí náhodné tajné číslo y, $1 \leq y \leq p-2$. Bob pošle Alici zprávu, obsahující hodnotu $\alpha^y \bmod p$.
4. Alice přijme $\alpha^y \bmod p$ a spočte tajný klíč k: $k = (\alpha^y)^x \bmod p$.
5. Bob přijme $\alpha^x \bmod p$ a spočte tajný klíč k: $k = (\alpha^x)^y \bmod p$, tj. získá stejnou hodnotu tajného klíče.

Oskar zná p a α z první zprávy, $\alpha^x \bmod p$ a $\alpha^y \bmod p$ ze druhé a třetí zprávy. Aby mohl spočíst spočíst tajný klíč $\alpha^{xy} \bmod p$, potřeboval by znát x a y; určit x z $\alpha^x \bmod p$ je obtížný problém.

Poznámka (k tvrzení „α musí splňovat ještě další podmínky“)

> Číslo α je generátor grupy Z_p^*, tj. jestliže mocním generátor, výsledná hodnota mi „obejde“ všechny prvky grupy. Generátor grupy se hledá heuristickými metodami.
> **Příklad:** Grupa $Z_5^* = \{1,2,3,4\}$ řádu 4 je cyklická a má generátor 2 (tj. pro všechna $a \in \{1,2,3,4\}$ existuje i takové, že $a = b^i$).
>
> ```
> 2^1 == 2 (mod 5) 2^2 == 4 (mod 5) 2^3 == 3 (mod 5) 2^4 == 1 (mod 5)
> 2^5 == 2 (mod 5) 2^6 == 4 (mod 5) ....
> ```
>
> □

Výše uvedený protokol poskytuje ochranu před pasivním odposlechem, není však odolný proti aktivnímu oponentovi. Oskar může přijmout zprávu (2) protokolu, hodnotu α^x nahradit nějakým $\alpha^{x'}$ a takto modifikovanou zprávu předat Bobovi. Pokud Oskar stejně modifikuje i zprávu (3), Alice po skončení modifikovaného protokolu sdílí s Oskarem klíč $\alpha^{xy'}$ a Bob klíč $\alpha^{x'y}$. Oba účastníci pak komunikují s Oskarem, který jejich zprávy přešifruje a předá druhé straně. Tento typ napadení se nazývá útok **man-in-the-middle**.

```
Autentizace s využitím asymetrické kryptografie
-----------------------------------------------

* předpokládejme, že Alice a Bob znají navzájem své veřejné klíče (což není
  tak jednoduché jak se zdá)
* pokud chtějí ustanovit relaci, mohou použít např. následující protokol:

1. Alice zašifruje svou identitu A a nonci r_A Bobovým veřejným klíčem E_B a
   pošle Bobovi zprávu:  E_B(A, r_A)
2. Bob rozšifruje, vygeneruje vlastní nonci r_B a relační klíč K a pošle
   Alici zprávu zašifrovanou veřejným klíčem Alice:  E_A(r_A, r_B, K)
3. Alice dešifruje, zkontroluje r_A a ví, že odpověď je od Boba; zašifruje
   r_B relačním klíčem K a pošle Bobovi:  K(r_B)
4. Po obdržení zprávy Bob ví, že komunikuje s Alicí.

Co může dělat Oskar? Může vytvořit zprávu (1), ale Alice dostane zprávu
obsahující r_A které nevygenerovala, takže nebude pokračovat. Zprávu (3)
Oskar vygenerovat neumí, protože nezná r_B ani K.

* protokol předpokládá, že Alice a Bob znají navzájem své veřejné klíče
* pokud předpoklad neplatí, musí si je předat

* např. pokud si je navzájem pošlou otevřeným kanálem, může Oskar zachytit
  zprávu E_A Alice pro Boba a místo ní poslat vlastní veřejný klíč E_o, totéž
  pro zprávu Boba pro Alici => oba komunikují prostřednictvím Oskara
 - Alice chce poslat zprávu Bobovi, zašifruje pomocí E_o, Oskar rozšifruje a
   zašifruje pomocí E_B a pošle Bobovi

* Rivest a Shamir (1984) navrhli tzv. interlock protokol, který řeší:

1. Alice zašifruje zprávu x: E_B(x), Bobovi pošle pouze sudé bity E_B(x)
2. Bob zašifruje zprávu y: E_A(y), Alici pošle pouze sudé bity E_A(y)
3. Alice pošle liché bity E_B(x), Bob zkombinuje a rozšifruje
4. Bob pošle liché bity E_A(y), Alice zkombinuje a rozšifruje

Ve výše zmíněné situaci vidí Oskar sudé bity E_o(x), nemůže dešifrovat i když
zná D_o(x) a proto nemůže předat Bobovi E_B(x).


Digitální podpisy
=================

* potřebujeme systém, kde jedna strana může posílat druhé "podepsané" dokumenty
* digitální podpisy by měly mít následující vlastnosti:
```

- příjemce může ověřit identitu odesilatele zprávy
- odesilatel nemůže později odmítnout obsah zprávy
- příjemce nemůže změnit obsah podepsané zprávy

Digitální podpisy založené na symetrické kryptografii
---

* předpokládejme, že máme centrální autoritu, která všechno ví a každý jí důvěřuje; obvykle se nazývá Velký bratr (Big Brogrer, zkratka BB)
* každý uživatel sdílí tajný klíč s BB; Alice sdílí K_A a Bob K_B

Alice chce Bobovi poslat podepsanou zprávu P:

1. Alice pošle BB zprávu: A, K_A(B, r_A, t, P)
2. BB dešifruje a pošle Bobovi zprávu: K_B(A, r_A, t, P, K_BB(A, t, P))

Pokud Oskar by mohl přehrát původní zprávu; staré zprávy budou odmítnuty na základě časového razítka t, novější na základě duplikovaného r_A (tj. Bob nemusí uchovávat všechna r_A).

Bob ví, že zpráva pochází od Alice, protože BB přijme od Alice pouze zprávu zašifrovanou K_A. Pokud Alice později odmítne zprávu jako podvrženou, může se Bob obrátit na KK s prosbou o rozšifrování K_BB(A, t, P); výsledek usvědčí Alici ze lži.

* nevýhoda - všechny strany musejí důvěřovat BB, BB si může přečíst všechny podepsané zprávy

❇