Správa klíčů (key management)

významná část bezpečnostní strategie nad danou doménou

Základním úkolem správy klíčů je kontrola klíčového materiálu po celou dobu jeho existence, zabraňující neautorizovanému odhalení, modifikaci, substituci, opakovanému nebo nesprávnému použití.
Důležitým úkolem správy klíčů je zajistit, že klíče nikdy nebudou přístupny v otevřené formě.
Obecné požadavky na správu klíčů:

- *Utajení dat* - tajné klíče a pravděpodobně i další informace je nutné podržet po dobu jejich přenosu nebo uložení v tajnosti.

- *Detekce modifikací* - je třeba vytvořit prostředky umožňující zabránit, nebo alespoň detekovat, možný neautorizovaný zásah do utajených či důvěryhodných informací.
- *Detekce opakovaného použití, časová razítka* - systém musí být odolný proti neautorizované duplikaci zpracovávaných dat, časová razítka zajišťují, že útočník nemůže jako odpověď na žádost podsunout již dříve zachycenou zprávu.

- *Autentizace entit* - je nutné ověřit, že entita je skutečně tou entitou, za kterou se vydává.

- *Autentizace původu dat* - je nutné ověřit, že zdroj dat je skutečně tím, za který se vydává.

- *Důkaz přijetí* - odesílatel musí mít možnost zjistit, zda odesílaná data byla v pořádku doručena adresátovi.

Nebezpečí prozrazení klíče roste s dobou a intenzitou jeho používání. Časté změny klíčů neúnosné.

Řešením je provádět přenos klíčů automaticky -> používány klíče pro šifrování klíčů (též sekundární klíče).
Nejvyšší vrstva klíčů - tzv. master klíče - je potom distribuována manuálně.

Služby správy klíčů:

Registrace entit - mechanismus, prostřednictvím kterého je pro systém prováděna autentizace uživatelů a zařízení

- Absolutní autentizace - poskytuje vazbu mezi formálním označením a fyzickou reprezentací dané entity.
- Relativní autentizace - umožňuje provést novou identifikaci entity s jistým formálním označením bez nutnosti spojovat ji s určitou reprezentací.

Generování klíčů - nepredikovatelný generátor náhodných čísel, pro aplikace vyžadující nejvyšší stupeň bezpečnosti nutné generátory pracující na skutečně náhodném, zvnějšku neovlivnitelném jevu.

Tvorba certifikátů - certifikáty používány pro účely autentizace. Autorita provádějící certifikaci doplněním certifikátu k danému objektu prohlásí tento za "pravý".
on-line / off line

Autenticita, verifikace - rozlišujeme tři druhy

- identifikace entit
- autenticita obsahu zprávy
- autenticita původu zprávy

Verifikace se týká samotného procesu dokazování určitých tvrzení, časté použití certifikátů, ověřování pravosti certifikátů.

Distribuce klíčů - procedury, kterými jsou klíče doručovány entitám, jež o ně legitimně žádají.

Dnes módní modulární systém správy klíčů, kde jednotlivé elementy protokolu splňují jim odpovídající požadavky:

- *Šifrování* - používají se běžné (a)symetrické algoritmy nebo speciální protokoly umožňující ustanovení sdíleného klíče.

- *Kódy pro detekci modifikací* - přidáním netriviálně odvoditelné redundance je možno ověřit, zda nedošlo ke změně přenášených dat. Používají se vhodné hašovací funkce, autentizační kódy zpráv (MAC), a různé MDC kódy.

- *Kódy pro detekci opakovaného použití* - používají se časové známky, čítače zpráv, jejichž obsah se kombinuje s klíčem či částí přenášených dat, ofsetování klíčů apod.

Údržba klíčů - aktivace klíčů, problematika úschovy klíčů, výměny klíčů, obnovy poškozených klíčů, černé listiny vyzrazených klíčů, deaktivace klíčů a jejich likvidace.

Služby často sdružovány do tzv. *zařízení správy klíčů* (key management facility), jejichž obsah je (fyzicky) chráněn proti poškození, vyzrazení, záměně, modifikaci atp.

Příklady: zařízení pro generování klíčů
zařízení pro přechovávání klíčů
zařízení pro autentizaci.
akreditační zařízení (enrollment facillity) - vydávání kredenciálů, tiketů

Z důvodu dosažení potřebné granularity popisu celého systému je možné i zařízení dále slučovat do větších celků - *jednotek správy klíčů* (key management units), nebo též serverů.

- certifikační autority (certification authority)
- centra pro distribuci klíčů (key distribution centres) atd.

Servery považovány za *důvěryhodné* (trusted)

- funkční důvěryhodnost - certifikační servery
- bezpodmínečná důvěryhodnost- servery poskytující šifrovací klíče

Point-to-point distribuce klíčů - v případě symetrických algoritmů nutná předchozí existence sdíleného klíče, v případě asymetrických systémů existence páru klíčů na každém uzlu a notarizované veřejné klíče na klíč-serveru

Jakákoliv distribuce klíčů založená na použití deterministických kryptografických algoritmů, **ne**umožňuje vyloučit nutnost provést jistou část tohoto procesu manuálně.

Správa klíčů pro asymetrické systémy
Správa klíčů pro symetrické systémy

Výměna exponenciálních klíčů (D-H).

Autory jsou Diffie a Hellman
Metoda je založena na obtížnosti počítání logaritmů nad GF(p)

1. Uzel A vypočítá

$$Y_a = \alpha^{X_a} \bmod p$$

   a pošle je druhé straně.

X_a - náhodně zvolené číslo, α - některý z generátorů GF(p).

2. Obdobnou akci provede i uzel B Y_b.
3. Obě strany spočtou

$$K_{ab} = \alpha^{X_a X_b} \bmod p$$

Případný útočník by musel K_{ab} spočítat pouze s použitím Y_a nebo Y_b což nejde rychleji, než spočítat logaritmus jednoho z čísel X_a nebo X_b.
Má-li p okolo 300 cifer, A i B potřebují řádově tisíc operací, potenciální útočník zhruba 10^{30} operací.
Dnes minimálně 2048 bitů (dále viz RSA, El-Gamal)
V reálném světě jsou nutné certifikáty veřejných klíčů
Pro dosažení lepší bezpečnosti se používá tzv. prchavý (ephemeral) mód – komunikující generují svůj pár klíčů pro ustavení šifrovacího klíče náhodně, autenticita veřejného klíče je zajištěna podpisem pomocí stálého páru klíčů, na který má uživatel certifikát autority

Distribuce klíčů založená na identifikaci komunikujících stran

Schéma založeno na obtížnosti výpočtu logaritmu nad vhodným okruhem, lze je rozšířit i o digitální podpisy a provádění identifikace uživatelů.

Inicializace

Centrum pro autentizaci klíčů (CAK) vybere 'one-way' funkci f, velké prvočíslo p a prvek α okruhu, který není triviálním dělitelem nuly. Všechna tato čísla zveřejní. P mělo být vybráno tak, aby mělo tvar $p = 2p' - 1$, kde p' je rovněž prvočíslo.

CAK zvolí svůj tajný klíč, náhodné číslo x z množiny $\{1,\ldots, p\text{-}1\}$ tak, aby **NSD**$(x, p\text{-}1) = 1$. Na základě tohoto čísla CAK spočítá svůj veřejný klíč

$$Y = \alpha^x \pmod p.$$

Registrace uživatele

Každá nová entita musí navštívit CAK se svojí identifikací ID_i a obdrží tzv. rozšířenou identifikaci EID_i jako

$$EID_i = f\left(ID_i\right)$$

a podpis (r_i, s_i) jež SAK vypočte dle vztahu

$$s_i = \left(EID_i - k_i r_i\right)x^{-1} \bmod p - 1.$$

Zde $r_i = \alpha^{k_i} \bmod p$ a k_i je náhodně zvoleno z množiny $\{1,\ldots ,p\text{-}1\}$, s_i je v systému používáno jako tajný klíč entity E_i.

Distribuce klíče

E_i a E_j chtějí sdílet společný klíč $k_{i,j}$. Entita E_i zašle (ID_i, r_i) entitě E_j a naopak. Entita E_i potom může spočítat klíč $k_{i,j}$ dle vztahu

$$k_{i,j} = \left(\alpha^{EID_j}\left(r_j^{r_j}\right)^{-1}\right)^{s_i} \bmod p = \left(Y^{s_j}\right)^{s_i} \bmod p$$

obdobně entita E_j může spočítat klíč $k_{j,i}$

$$k_{j,i} = \left(\alpha^{EID_i}\left(r_i^{r_i}\right)^{-1}\right)^{s_j} \bmod p = \left(Y^{s_i}\right)^{s_j} \bmod p$$

Protože $k_{i,j} = k_{i,j}$, lze tyto hodnoty použít jako sdíleného klíče pro entity E_i a E_j. Popsané schéma umožňuje pouze nepřímou autentizaci klíčů v tom smyslu, že použití správného klíče lze poznat pouze dle smysluplnosti dešifrovaného textu.

Analýza

Předpokládá se odolnost vůči nejznámějším útokům, protokol je z hlediska možného rozbití patrně ekvivalentní s výpočtem diskrétního logaritmu.
Výhodou protokolu je možnost unifikovaného rozšíření o autentizaci zpráv včetně časových známek a identifikace uživatelů.

Systém distribuce klíčů pro konference komunikujících entit

schopnost vytvořit společný sdílený klíč pro libovolnou skupinu komunikujících - konferenci.

Protokol

umožňuje ustanovit sdílený tajný klíč bez předchozí komunikace jednotlivých členů zamýšlené konference, autentizaci informací, předávaných v rámci procesu ustanovování klíče. Opět je nezbytná existence důvěryhodného centra.

Krok 1: Centrum vygeneruje tři velká prvočísla p, q, r a číslo $n = pq$. Dále nalezne dvojici (e, d) způsobem podobným, jako v kryptosystému RSA tak, aby platilo

$$ed \equiv 1 \pmod{L},$$

$$L = \mathrm{NSD}\big((p-1),(q-1),(r-1)\big), \quad 3 \le e, d < L$$

Rovněž určí prvočíslo c ($3 <= c < L$) a číslo g, jež je primitivním prvkem v Galoisových okruzích $GF(p)$. $GF(q)$ a $GF(r)$. Pro každou entitu i mající identifikaci I_i centrum spočítá

$$s_i = I_i^d \bmod nr$$

Na závěr tohoto kroku entita i obdrží šestici (n, r, g, e, c, S_i).
Pokud již neočekáváme přijetí žádných dalších entit, je možné čísla p, q a d zrušit. Číslo S_i si každá entita podrží v tajnosti, zbylá čísla jsou společná pro všechny.

Krok 2: V rámci tohoto kroku probíhá vlastní tvorba sdíleného klíče konference.

1. Entita *j* vygeneruje náhodné číslo U_j které uchová v tajnosti a všem zašle E_j, kde $$E_j = g^{eU_j} \bmod n .$$
2. Entita *i* vygeneruje náhodné číslo P_i, které není dělitelem *r*-1 a spočítá $\overline{P_i}$ tak, aby platilo $P_i \overline{P_i} \equiv 1 \ \left(\mathrm{mod}\left(r-1\right)\right)$. Obě čísla ponechá v tajnosti, stejně jako náhodně vygenerované číslo V_i. Entitě *j* odešle čtveřici (X_i, Y_i, Z_{ij}, F_i):
$$X_i = g^{eP_i} \bmod nr \ , \ Y_i = S_i g^{cP_i} \bmod nr \ ,$$
$$Z_{ij} = E_j^{P_i} \bmod nr \ , \ F_i = X_i^{eV_i} \bmod n .$$
3. Entita *j* obdrží (X_i, Y_i, Z_{ij}, F_i) a ověří platnost následujících kongruencí:
$$Y_i^e \left/ X_i^c \equiv I_i \ \left(\mathbf{mod}\, nr\right), \ Z_{ij} = X_i^{U_j} \ \left(\mathrm{mod}\, n\right)\right.$$
Pokud se podaří ověřit platnost uvedených kongruencí, entita *j* verifikovala, že odesílatelem přijatých dat je opravdu entita *i*. Entita *j* tedy vygeneruje náhodné číslo R_j, které rovněž podrží v tajnosti. Entitě *i* zašle trojici (A_{ji}, B_{ji}, C_{ji}), kde
$$A_{ji} = X_i^{eR_j} \bmod nr \ , \ B_{ji} = S_j X_i^{cR_j} \ \mathbf{mod}\, nr \ ,$$
$$C_{ji} = F_i^{R_j} \bmod n$$
přičemž si ponechá $A_{jj} = X_j^{eR_j}$.
4. Entita *i* přijme (A_{ji}, B_{ji}, C_{ji}). Ověřením platnosti následujících kongruencí zjistí, zda přijatá data byla opravdu zaslána entitou *j*:
$$B_{ji}^e \left/ A_{ji}^c \equiv I_j \ \left(\mathrm{mod}\, nr\right), \ C_{ji} \equiv A_{ji}^{V_i} \ \left(\mathrm{mod}\, n\right)\right.$$
Pokud je vše v pořádku, je entita *i* již schopna spočítat sdílený klíč zamýšlené konference dle vztahu
$$K = \left(\prod_{j=1}^{m} A_{ji} \right)^{\overline{P_i}} \bmod r$$
5. Získaný klíč je vskutku sdíleným tajným klíčem konference, neboť
$$K = g^{e^2\left(P_i R_1 + P_i R_2 + \ldots + P_i R_m\right)\overline{P_i}} \ \mathbf{mod}\, r = g^{e^2\left(R_1 + R_2 + \ldots + R_m\right)} \ \mathbf{mod}\, r$$

Analýza

Útok na jednotlivé části protokolu by měl být ekvivalentní s vyřešením alespoň jednoho ze dvou známých obtížných problémů - výpočet prvočíselného rozkladu, nebo výpočet logaritmu nad Galoisovým polem.

Správa klíčů pro bezpečnou komunikaci ATM terminálů

Systém umožňující uživatelům platby prostřednictvím bezhotovostních převodů. Každý uživatel má vlastní kartu obsahující jeho osobní údaje, číslo účtu atd. Každý z uživatelů má rovněž přidělen PIN konstantní délky.
Systém potom sestává z množství terminálů sloužících jako uživatelské rozhraní a několika center, které si lze představit jako pobočky různých bank.

Protokol

Klíč transakce

Klíč karty *Kk* - závislý na osobě majitele karty
Každý terminál obsahuje pro každé centrum *i*, se kterým komunikuje, jeden klíčový registr KR_i, jehož hodnota je využívána pro tvorbu klíče transakce.

Předp.systém v běžném provozním stavu, uživatel právě vložil kartu a dožaduje se platby.

1. Terminál přečetl *Kk* a číslo účtu *CU* jejího majitele a zjistil, které centrum provádí správu tohoto účtu.
2. Je vygenerován klíč transakce *KT:*

$$KT = \left(\mathbf{DEA}\left(Kk, KR_i\right)\right) \oplus Kk$$

3. Aby i centrum bylo schopno sestavit klíč, zpráva, kterou terminál centru posílá, obsahuje v otevřené podobě číslo účtu a identifikaci terminálu.
4. Centrum rovněž má k dispozici registr klíče, se shodnou hodnotou a dostatečné informace o každém spravovaném účtu, je schopno si ze zaslaných informací též sestavit klíč transakce.

Dotaz

1. Dotaz obsahuje ve formě otevřeného textu číslo účtu a identifikaci terminálu.

Uživatelovo PIN, požadovanou částku atd., je samozřejmě nutné separátně zašifrovat s použitím klíče transakce.

2. Autentizační blok zprávy je vygenerován procedurou pro tvorbu autentizačního bloku zprávy. Dotaz je šifrován použitím algoritmu DEA metodou CFB. Autentizační blok zprávy vznikne jako poslední výstupní blok tohoto procesu.

3. Autentizační blok rozdělíme na dvě poloviny.
Levá polovina bude sloužit jako autentizační kód.
Zbývajících 32 bitů autentizačního bloku označme jako reziduum dotazu *RD*. Terminál si jej uschová pro pozdější autentizaci odpovědi od centra.

4. Centrum po přijetí dotazu zkonstruuje klíč transakce, od dotazu oddělí autentizační kód, spočítá autentizační blok zprávy a ověří jej. Dále si ponechá pravou část reziduum dotazu pro pozdější použití.

Odpověď

1. Centrum sestaví odpověď a opět spočítá autentizační blok zprávy. Před počítáním autentizačního bloku centrum připojí na začátek zprávy *RD*.
2. Levá část autentizačního bloku je jako autentizační kód připojena ke zprávě a odeslána terminálu. Pravou část, označme ji reziduum odpovědi *RO*, si centrum ponechá.
3. Terminál obdobným způsobem jako v případě dotazu centrum vypočítá autentizační blok a provede verifikaci přijatých dat. Pokud je vše v pořádku, v souladu s došlou odpovědí provede odpovídající akci a dokončí svoji část transakce. I terminál si ponechá pravou část spočítaného autentizačního bloku - reziduum odpovědi.

Aktualizace registru klíče

1. Centrum zruší klíč transakce, uschová původní hodnotu registru klíče.
2. Nová hodnota registru klíče je vypočítána takto:

$$KR_i = \left(\mathbf{DEA}\left(RO + RD, KR_i\right)\right)$$

3. Terminál zruší klíč transakce a způsobem stejným jako centrum provede aktualizaci svého registru klíče.
4. Pokud se však ztratí odpověď od centra, terminál pro příští transakci použije tuto starou hodnotu registru klíče. Centrum registr klíče naplní uschovanou předchozí hodnotou a pokusí se znovu o autentizaci.

Analýza

Klíč transakce je ‘end-to-end’ a je jedinečný pro každou transakci. Znalost tohoto klíče transakce a klíče karty následující karty nemožní odvodit klíč následující transakce, ani něco o předešlé transakci.
Inicializaci a případnou reinicializaci je možno provést například v rámci instalace a nebo běžné údržby terminálu použitím speciální servisní karty.

Správa klíčů pomocí kontrolních vektorů

S každým klíčem *K* asociován kontrolní vektor *C* skládající se z množiny polí specifikujících přípustné použití klíče v rámci systému.
Klíč a odpovídající kontrolní vektor kryptograficky spojeny, aby byla vyloučena možnost následné změny vektoru.

Klíč nastavuje vlastní kryptografický algoritmus výběrem mapovací funkce, kontrolní vektor slouží k nastavování procesoru kryptografické jednotky - je takto vymezeno povolené použití daného klíče.
Každé kryptografické zařízení je navrženo tak, aby při pokusu o použití klíče nejprve provedlo verifikaci, zda tento pokus je v souladu s povoleným rozsahem operací, jak je zaznamenáno v kontrolním vektoru.

Kryptografické spojování K a C

Spojení *K* a *C* je nutné provést tak, aby nebylo možno provádět následné změny kontrolního vektoru.
Možnosti:

- zapojit kontrolní vektor do procesu šifrování a dešifrování klíče
- provést konkatenaci klíče a kontrolního vektoru a na vzniklém řetězci spočítat autentizační kód *AC*.

Nevýhodou druhé možnosti je nutnost stálého ověřování *AC* vždy při každém použití klíče.

Algoritmus pro šifrování a dešifrování klíčů

Vstupem algoritmu je kontrolní vektor *C*, klíč pro šifrování klíčů *KK* a klíč *K*.
C je hašovací funkcí **h** převeden na 128-bitovou hodnotu *H*, kterou je XOR-ován klíč *KK*. Následně je *K* zašifrován klíčem $KK \oplus H$ e-d-e algoritmem. Výsledkem

$$\mathbf{e}_{KK \oplus H}(K).$$

Hašovací funkce zajišťuje normalizaci délky kontrolního vektoru na 128 bitů.

Generování klíčů

Schéma obsahuje rovněž funkci **G** pro tvorbu nových klíčů s výstupem

$$\mathbf{e}_{key1 \oplus H1}(K) \text{ a } \mathbf{e}_{key2 \oplus H2}(K)$$

K - interně generovaný náhodný klíč, *C*1 a *C*2 kontrolní vektory a *key1* a *key2* vhodné klíče - např. master klíče jednotlivých zařízení, klíče pro šifrování klíčů sdílené mezi odesílatele a příjemce, nebo mezi toto generující zařízení a nějaké další zařízení v systému.
H1 a *H2* jsou hodnoty hašovací funkce pro oba kontrolní vektory.

Distribuce klíčů

Zadáme vstupní parametry **G** (KM_i, *C*1) a (KK_{ij}, *C*2), tedy na místě *key1* je použit master klíč zařízení *i* a na místě *key2* klíč pro šifrování klíčů sdílený mezi zařízení *i* a *j*. Výsledkem jsou balíky

$$\left(\mathbf{e}_{KM_i \oplus H1}(K), C1\right) \text{ a } \left(\mathbf{e}_{KK_{ij} \oplus H2}(K), C2\right)$$

První z nich je uložen v zařízení *i* a druhý je dopraven do zařízení *j*.
Zde je proveden import - došlý balík je dešifrován, přenesený klíč *K* je okamžitě znovu zašifrován master klíčem KM_j za použití *C*2 a uložen.

Kerberos

pracovní stanice považujeme za nedůvěryhodné, požadujeme autentizaci žadatele při každém požadavku na službu. Každý uživatel a každá služba má vlastní heslo. Systém obsahuje důvěryhodný autentizační server.

1. V rámci přihlašování do systému uživatel zadá své jméno.
2. Stanice zašle autentizačnímu serveru zprávu

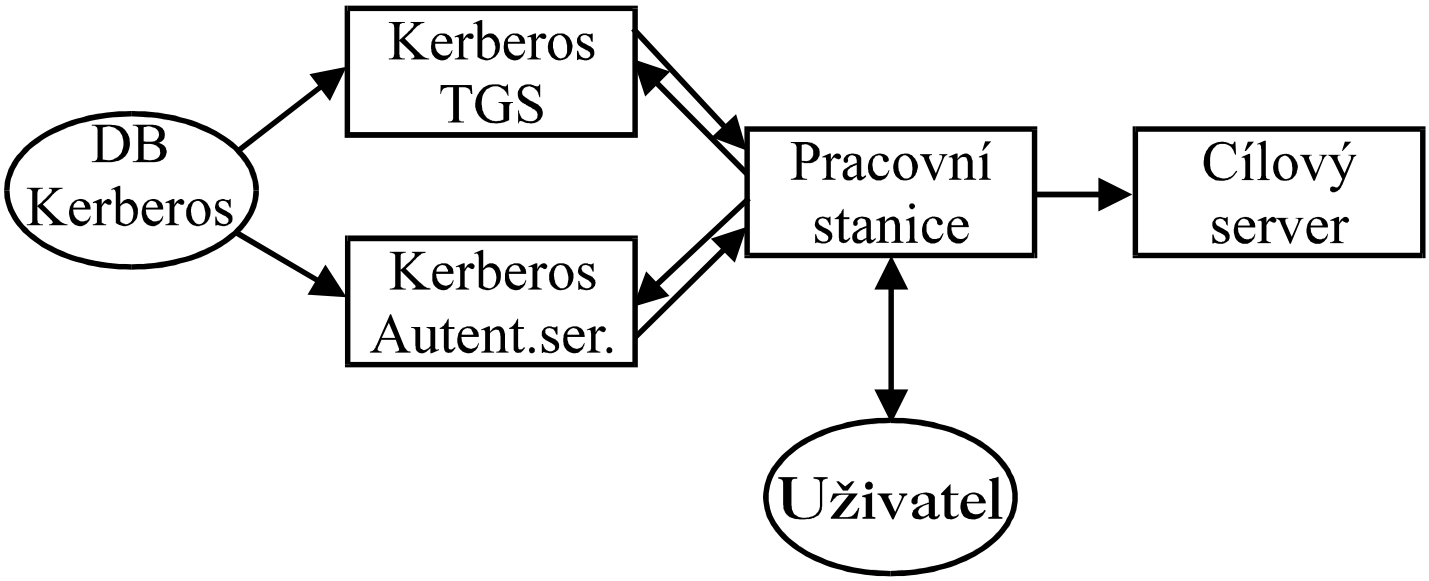
   {*login-name*, *TGS-name*}

   TGS-name - jméno akreditačního serveru.
3. Autentizační server vyhledá šifrovací klíč obou těchto entit. (klíči jsou zašifrovaná hesla)

4. Autentizační server vytvoří odpověď
   {*TGS-session_key, sealed_ticket*}
   zašifruje ji uživatelovým heslem a zašle stanici.
   Ticket má tvar:
   {*login-name*, *TGS-name, WS-net_address, TGS-session_key* }
   zašifrováním ticketu šifrovacím klíčem *TGS* vznikne *sealed_ticket. WS-net_address* je adresa stanice, *TGS-session_key* klíč označující relaci s *TGS*.
5. Po přijetí zprávy z bodu 4. stanice požádá uživatele o heslo, zašifruje je a získaný klíč použije k dešifrování zprávy a uschová ji.

Při žádosti o libovolnou službu bude nyní třeba nejprve získat odpovídající lístek.

1. Stanice zašle akreditačnímu serveru žádost
   {*sealed_ticket, sealed_authenticator, end_server_name*}
   kde *authenticator má tvar*
   {*login-name*, *WS-net_address, current_time*}
   Sealed_authenticator vznikne zašifrováním klíčem *TGS-session_key.*
2. Akreditační server dešifruje *sealed_ticket*, čímž získá *TGS-session_key.* Tímto klíčem dešifruje *sealed_authenticator,* ověří platnost lístku (*login-name, TGS-name, WS-net_address*) a porovná čas.
3. Akreditační server vyhledá heslo cílového serveru, a vytvoří klíč transakce *session-key*.
4. Pracovní stanici akreditační server zašle zprávu
   {*session_key, sealed_ticket*}
   kde *sealed_ticket* vznikne zašifrováním ticketu
   {*login-name*, *end_server_name, WS-net_address, session_key* }
   klíčem cílového serveru.
   Před odesláním je zpráva zašifrována klíčem *TGS-session_key.*
5. Dešifrováním zprávy z předchozího bodu stanice získá ticket pro cílový server. Rovněž od akreditačního serveru dostane nový *TGS-session_key*.
6. Stanice zašle cílovému serveru zprávu
   {*sealed_ticket, sealed_authenticator, end_server_name*}
   kde *sealed_authenticator* je
   {*login-name*, *WS-net_address, current_time*}
   zašifrovaný klíčem *session_key.*
7. Cílový server dešifruje *sealed_ticket* a následně *sealed_authenticator* a provede obdobnou verifikaci jako akreditační server v bodě 2. tohoto postupu.
8. Je-li vše v pořádku, vyhoví požadavku a odpověď zašifruje klíčem *session_key.*


Kerberos
TGS
DB
Kerberos
Pracovní
stanice
Cílový
server
Kerberos
Autent.ser.
Uživatel