پروتکلهای احرازاصالت

Authentication protocols

فهرست مطالب

- مقدمه
- احرازاصالت ضعیف(کلمات عبور)
- احرازاصالت قوی (مبتنی بر سؤال و جواب)
- پروتکل کربروس

مقدمه

- چرا احراز اصالت؟
  - حضور موجودیتهای گوناگون و پرشمار در شبکه
  - داده‌ها و سرویسهای مختلف باید در اختیار موجودیتهای خاص و شناخته‌شده قرار گیرند.
    - هر موجودیتی حق دسترسی به هر سرویس و داده‌ای را ندارد.
  - برخی موجودیتها ممکن است هویت برخی دیگر را جعل کنند.
    - دسترسی به سرویسها و اطلاعات حساس، پنهان کردن هویت موجودیت خرابکار، فرار از حسابرسی و ... .

مقدمه

- احراز اصالت موجودیتها: تأیید هویت مورد ادعای یک موجودیت در زمان معین مورد نظر
  - احراز اصالت یکطرفه
  - احراز اصالت دوطرفه
- عملیات احراز اصالت معمولاً با انجام تعدادی تبادلات رمزنگاری انجام می‌شود که پروتکل احراز اصالت نام دارد.

Basis for Authentication

- Something you **know** (a PIN, or password).
- Something you **have:**
  - secureID card or other token, generating a one-time password.
  - a key imbedded in a ‘secure area’
  - a smartcard (which may have keys imbedded and can perform cryptographic operations on behalf of a user).
- Something you **are** (a biometric).

فهرست مطالب

- مقدمه
- احرازاصالت ضعیف(کلمات عبور)
- احرازاصالت قوی (مبتنی بر سؤال و جواب)
- پروتکل کربروس

احراز اصالت ضعیف

- احراز اصالت ضعیف
- احراز اصالت یک طرفه
- انتخاب یک رشته از کاراکترها توسط کاربر به عنوان راز مشترک بین کاربر و سیستم
- وارد کردن کلمه عبور در هر بار احراز اصالت

احراز اصالت ضعیف

- **فایل حاوی متن ساده کلمات عبور**
  - ذخیره متن ساده کلمات عبور در سیستم
  - محافظت از فایل در برابر خواندن و نوشتن
  - غیر مبتنی بر رمزنگاری
  - ارسال فاش کلمه عبور بر روی خط ارتباطی کاربر و سیستم
  - امکان حمله تکرار
  - امکان سوء استفاده افراد دارای حق دسترسی

احراز اصالت ضعیف

- **فایل حاوی مقدار درهم کلمات عبور**
  - درهم سازی کلمات عبور و ذخیره نتیجه در سیستم
  - محاسبه مقدار درهم کلمه عبور وارد شده و ارسال آن بر روی خط ارتباطی کاربر و سیستم
  - محافظت از فایل فقط در برابر نوشتن
  - امکان حمله تکرار

Password Vulnerabilities

- Writing them down
- Stolen passwords (via eavesdropping)
  - Trojan Horse
- Poor password choice
  - Easy to guess, easy to remember
  - People use the same password multiple times
  - Passwords changed infrequently
- Offline attacks
  - Search through password dictionary

Survey of 3,289 Passwords

- With no constraints on choice of password
  - 15 were a single ASCII letter.
  - 72 were strings of two ASCII letters.
  - 464 were strings of three ASCII letters.
  - 47 were strings of four alphanumerics.
  - 706 were five letters, all upper-case or all lower-case.
  - 605 were six letters, all lower case.

حملات علیه روشه های مبتنی بر کلمه عبور و راه کار مقابله

- جستجوی کامل
- جستجوی برخط. روش های مقابله:
  - اطمینان از انتخاب کلمات عبور از بین یک فضای بزرگ
  - محدودیت گذاشتن روی تعداد دفعات وارد کردن نا موفق کلمه عبور در یک بازه زمانی معین
  - کاهش سرعت فرآیند نگاشت و یا وارد کردن کلمه عبور
- جستجوی برون خط
  - استفاده از فایل حاوی مقدار درهم کلمات عبور
  - وابستگی موفقیت حمله به تعداد کلمات عبور، نوع نگاشت آنها و میزان پردازنده در اختیار (برای موازی سازی)

حملات علیه روشه های مبتنی بر کلمه عبور و راه کار مقابله

- حمله تکرار
  - شنود خط ارتباطی
  - مشاهده هنگام تایپ یا احتمالا جایی که نوشته شده
- حملات حدس کلمه عبور و واژه نامه
  - استفاده از واژه نامه های مرسوم، برخط و یا تخصصی
  - انتخاب کلمه عبور از یک فضای بسیار کوچک

بهبود امنیت در روش های مبتنی بر کلمه عبور

- قوانین کلمه عبور
- کاهش سرعت نگاشت کلمه عبور
- نمک زدن به کلمات عبور
- استفاده از عبارات عبور
- سازوکارهای بازدارنده

نمک زدن

- افزودن یک رشته تصادفی t بیتی (نمک) به کلمه عبور و سپس اعمال تابع درهم ساز بر آن
- ذخیره فاش نمک به همراه مقدار درهم کلمه عبور
- دشوار کردن حملات واژه نامه
- عدم تاثیر بر عملکرد حدس کلمه عبور یک کاربر مشخص

عبارتهای کلمه عبور

- وارد کردن یک عبارت به جای یک کلمه
- افزایش آنتروپی با زیاد کردن تعداد کاراکترها
  - آنتروپی یک کاراکتر در یک کلمه یا عبارت عبور در مقابل یک رشته تصادفی
- نیاز به تایپ اضافی
  - وقت گیر بودن
  - افزایش احتمال اشتباهات تایپی

کلمه عبورهای یک بار مصرف

- فقط یکبار استفاده از هر کلمه عبور
- مقاوم در برابر شنود به منظور جعل هویت بعدی
- سه روش اصلی پیاده سازی
  - استفاده از لیست های از پیش مشترک کلمات عبور
    - نگهداری امن لیست در دو طرف
    - حفظ ترتیب
  - استفاده از کلمات عبوری که متناوباً به روز می شوند
    - توافق روی کلمه عبور بعدی در هر ارتباط
    - ایجاد مشکل در صورت قطع ارتباط

کلمه عبورهای یک بار مصرف

- دنباله های کلمه عبور یکبار مصرف مبتنی بر توابع یک طرفه
  - کاراتر از دو روش قبل
  - نوعی پروتکل چالش- پاسخ با تعیین غیرمستقیم چالش توسط موقعیت جاری در دنباله
  - مثال: شمای کلمه عبور یکبار مصرف Lamport

کلمه عبورهای یک بار مصرف

- انتخاب راز w، ثابت t و ارسال $w_0=H^t(w)$ برای سیستم توسط کاربر
- مقداردهی $i_A=1$ توسط سیستم
- iامین بار احراز اصالت $A \to B: A, i, w_i \ (= H^{t-i}(w))$
- بررسی $i=i_A$ و $H(w_i)=w_{i-1}$ توسط سیستم
- افزایش شمارنده و ذخیره w_i برای نشست بعدی

W
H(-)

$\{A, i, H^{(t-i)}(w)\}$

$H^{(t-i+1)}(w)$,
H(-)

کلمه عبورهای یک بار مصرف

- انتخاب راز اولیه p توسط کاربر و ارسال آن برای کارگزار
- ارسال (r, H(r, p)) در هر مرحله احراز اصالت
  - r یک مقدار تصادفی، مهر زمانی و یا شماره ترتیب
- امکان جعل هویت به وسیله دستبرد به اطلاعات کارگزار
  - ضعف این روش نسبت به پروتکل Lamport

کلمه عبورهای یک بار مصرف

- ضعف در برابر دشمن فعالی که پیام کاربر را سد کرده و بعداً برای کارگزار ارسال می کند
- همچنان امکان اجرای حمله حدس کلمه عبور با استفاده از پیام های مبادله شده
- راه حل: استفاده از پروتکل های چالش- پاسخ مبتنی بر کلمه عبور

احراز اصالت دو عاملی

- شماره شناسایی شخصی
- استفاده به همراه یک توکن
  - وارد کردن PIN هنگام استفاده از توکن
  - ایجاد سطح دوم امنیت هنگام گم یا دزدیده شدن توکن
- کوتاه و شماره ای
  - ۴ تا ۸ رقم
  - امکان جستجوی کامل
  - توقیف و یا غیرفعال کردن توکن در صورت چند بار اشتباه وارد کردن PIN

فهرست مطالب

- مقدمه
- احرازاصالت ضعیف(کلمات عبور)
- احرازاصالت قوی (مبتنی بر سؤال و جواب)
- پروتکل کربروس

Strong Authentication

- In **strong** authentication, one entity ‘proves’ its identity to another by demonstrating knowledge of a secret known to be associated with that entity, *without revealing that secret itself during the protocol.*
- Also called `challenge-response’ authentication.
- Use cryptographic mechanisms to protect messages in protocol:
  - Encryption.
  - Integrity mechanism (e.g. MAC).
  - Digital signature.

Encryption-based Unilateral Authentication

1. A → B: ‘Hi Bob, I’m Alice’
2. B → A: R (challenge)
3. A → B: $\{R \,||\, B\}_K$ (response)

(Here, $\{X\}_K$ means string X encrypted under key K, and || means concatenation of strings.)

Security of the Protocol

- Eve ‘sees’ R and $\{R \,||\, B\}_K$. Because of idealised encryption, she should learn nothing about K.
- Bob gets his challenge R back again, in an encrypted form that **only** Alice can prepare. This allows him to be sure of origin and integrity of the message.
- But Mallory can impersonate Bob easily: so Bob not authenticated to Alice. Only unilateral authentication (of Alice to Bob).

Replay Attack

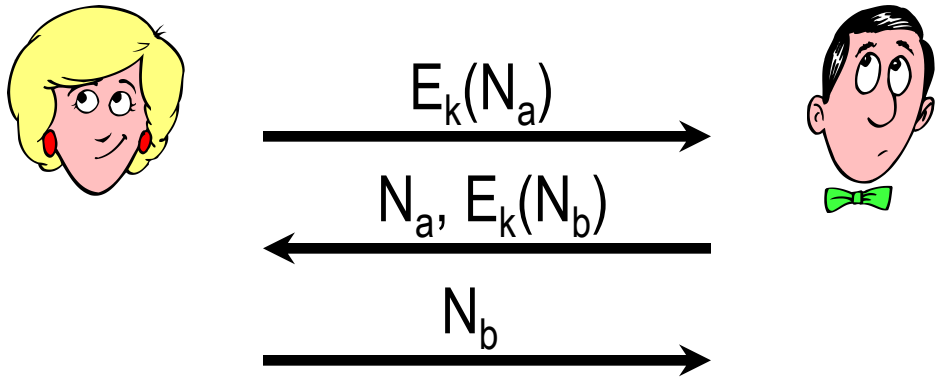
- Mallory can't prepare the correct response $\{R \| B\}_K$ to Bob's challenge because he doesn't know K.
- But R must be unpredictable: otherwise Mallory can masquerade as Alice in a subsequent protocol run, *replaying* Alice's response.

Replay Attack

- The replay attack shows that origin and integrity checking are not enough – we also need a means of checking *freshness* of messages and *liveness* of principals.
- **Freshness:** assurance that message has not been used previously and originated within an acceptably recent timeframe.
- **Liveness:** assurance that message sent by a principal within an acceptably recent timeframe.
- Three main methods for providing freshness:
  - Nonce (Number used ONCE).
  - Sequence numbers
  - Time-stamps (clock-based or `logical' time-stamps).

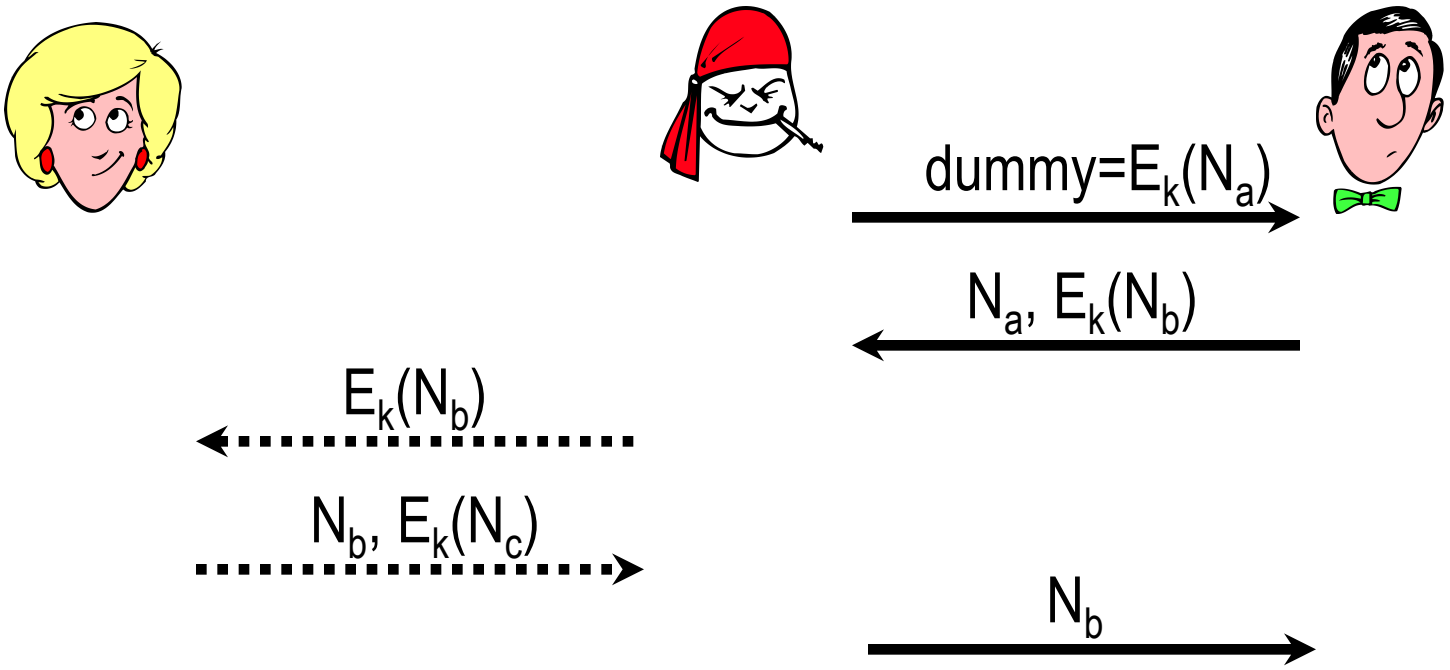
Standard bilateral Authentication

- Alice and Bob share (strong) key k
- Simple challenge-response type protocol: (N_a,N_b - nonces)



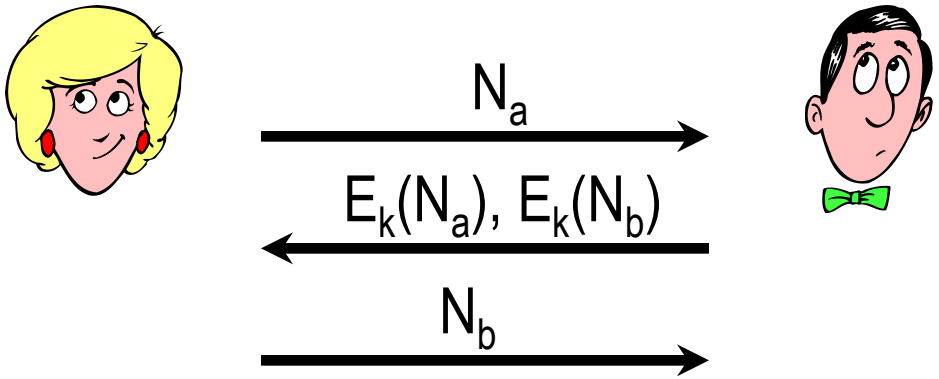
Attack on Simple Protocol

- "Oracle Attack"



Protocol Fix?

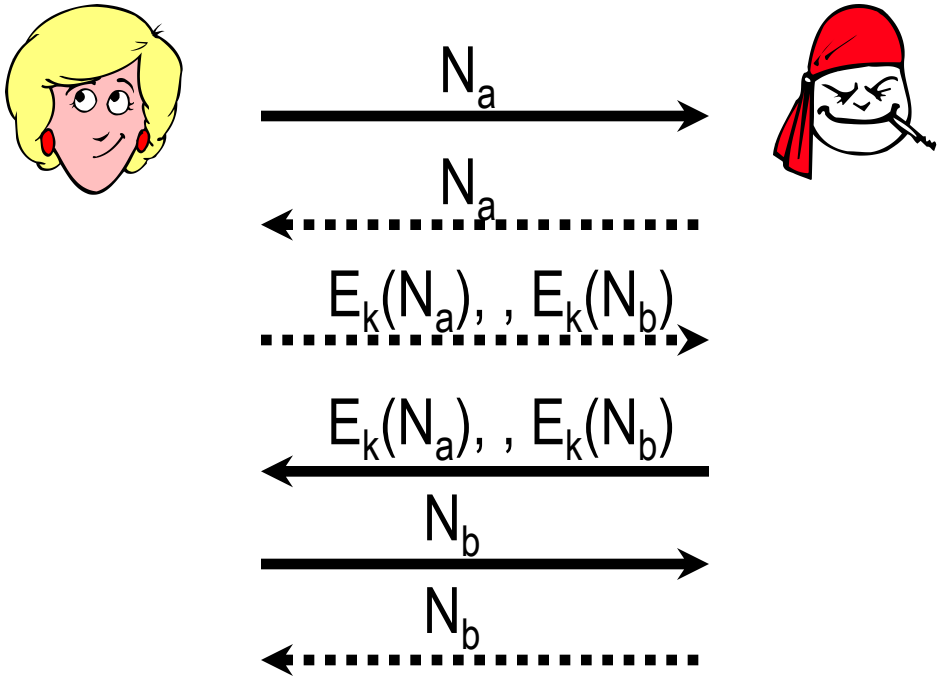
- Modified challenge-response type protocol:



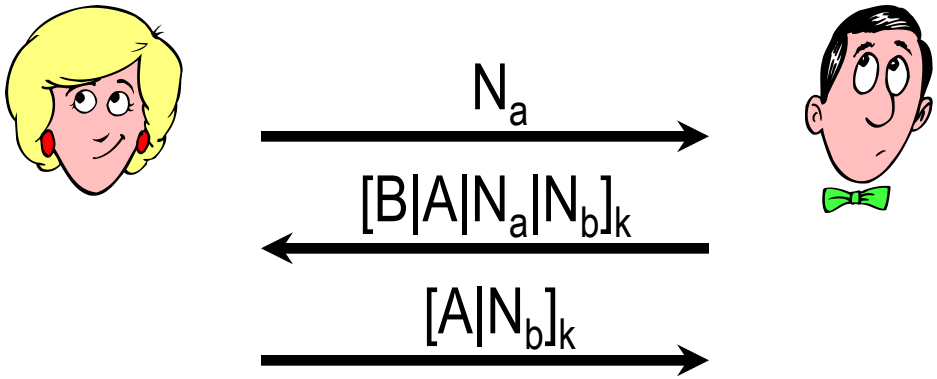
این پروتکل نیز آسیب پذیر است

Attack on Modified Protocol

- "Parallel Session Attack"



Correct Protocol



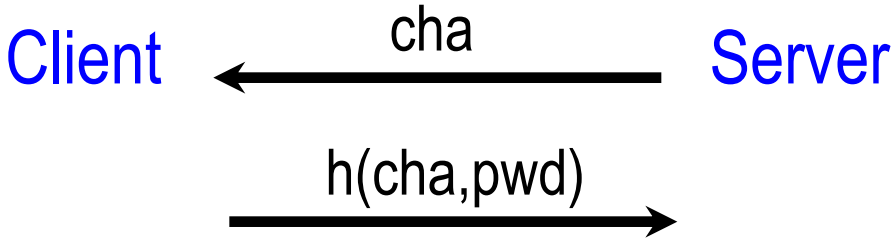
Password-based Protocols

- Telnet - vulnerable to replay attacks

- Hashing does not help

Password-based Protocols

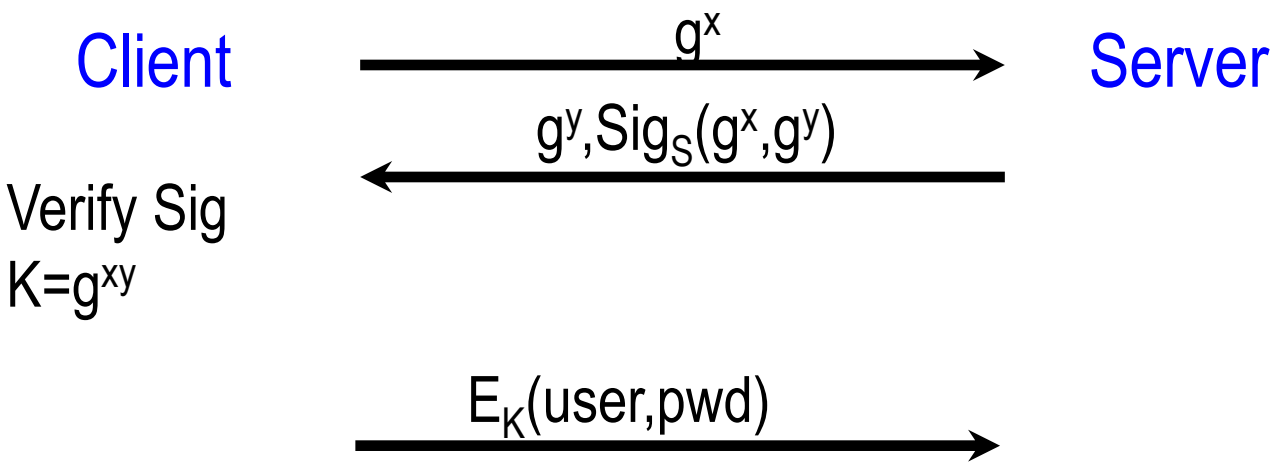
- Challenge-Response : vulnerable to offline dictionary attacks



- Problem: “verifiable text”

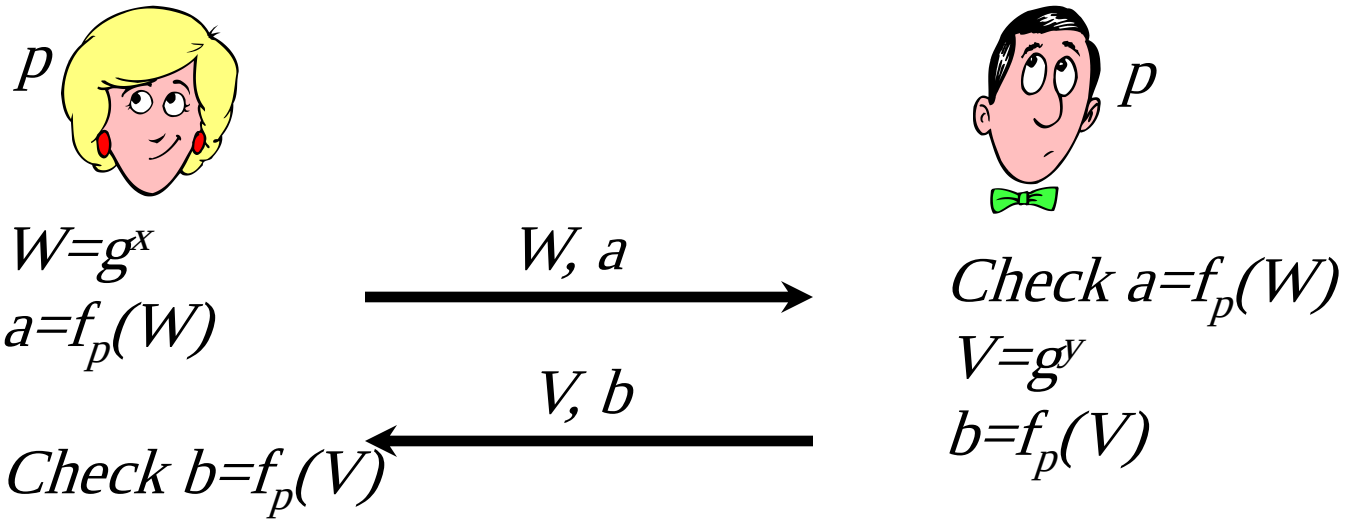
Password-based Protocols

- SSH: Relies on public key



- Similar protocols relying on public keys:
  - [SSL],[Halevi-Krawczyk],[Boyarsky],[Shoup]

Password Auth. - Attempt



- **Intuition:** authenticate Diffie-Hellman values using PRF with password as key.
- **Insecure!** (eavesdropper obtains verifiable text)

Kerberos

کربروس

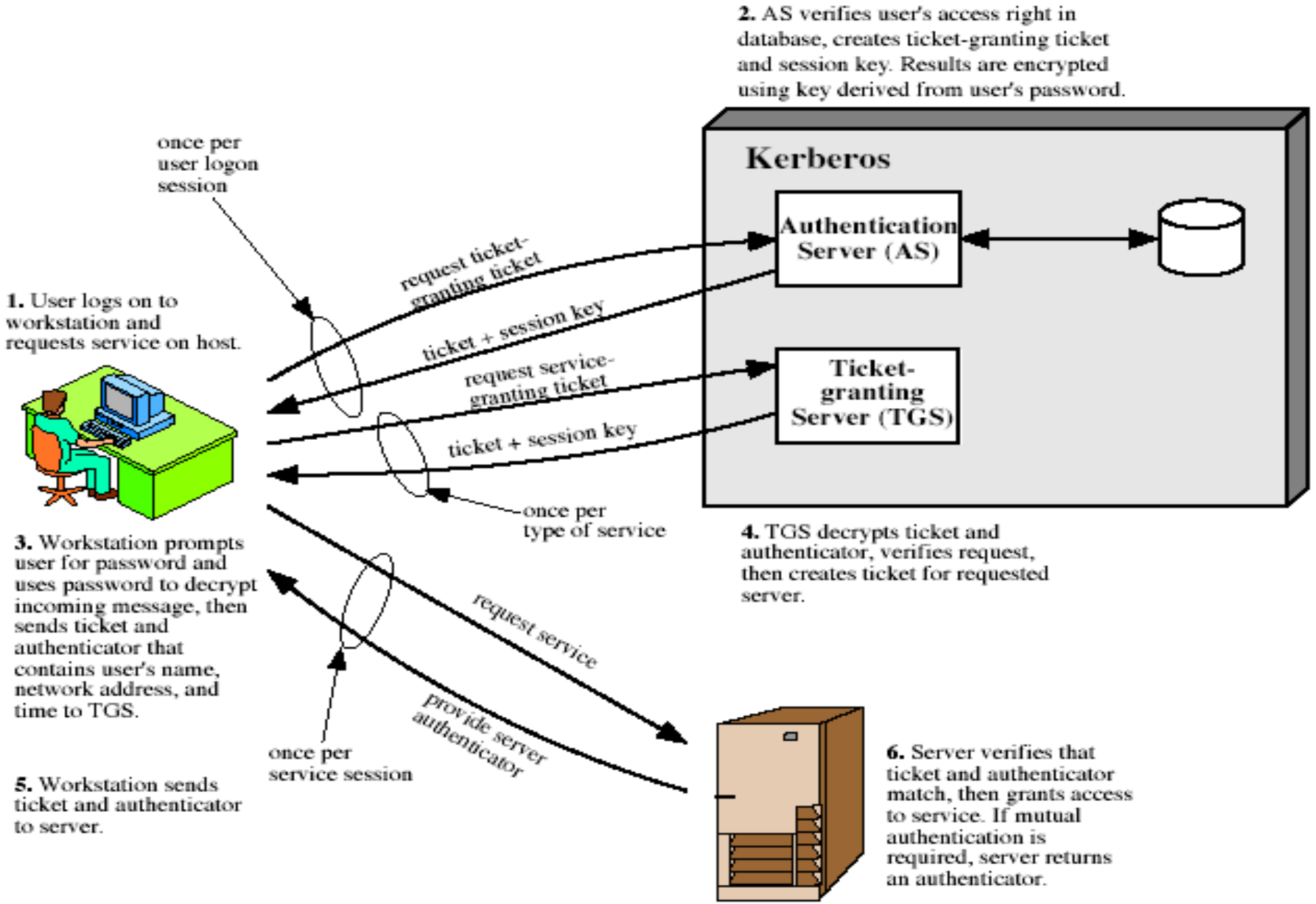
- پروتکل احراز هویت بر اساس رمز نگاری کلید متقارن
- طراحی شده در MIT
- به جای احراز هویت در هر کارگزار به صورت توزیع شده، یک کارگزار خاص را به احراز هویت اختصاص میدهیم
- نسخه های ٤ و ٥ آن در حال استفاده هستند
- احراز هویت دو جانبه (mutual) برقرار میشود.
  - کارگزاران و کارفرمایان هردو از هویت طرف مقابل اطمینان حاصل میکنند

ویژگیهای عمومی کربروس

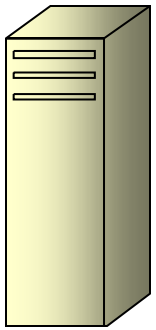
چند تعریف

- دامنه: یک محدوده دسترسی را مشخص می کند. به نوعی معادل دامنه های تعریف شده در ویندوز می باشد.
- مرکز توزیع کلید: معادل کارگزار کربروس می باشد.
- Principal : به سرویس ها، دستگاه ها، کاربران و کلیه عناصری که احتیاج به شناساندن خود به کارگزار کربروس دارند، گفته می شود.
- بلیط: در واقع نوعی گواهی است که هنگام ورود کاربر به قلمرو کربروس به او داده می شود که بیانگر اعتبار او برای دسترسی به منابع شبکه می باشد.

کربروس نسخه ٤: شمای کلی



کربروس نسخه ٤: بررسی الگوریتم-١



AS

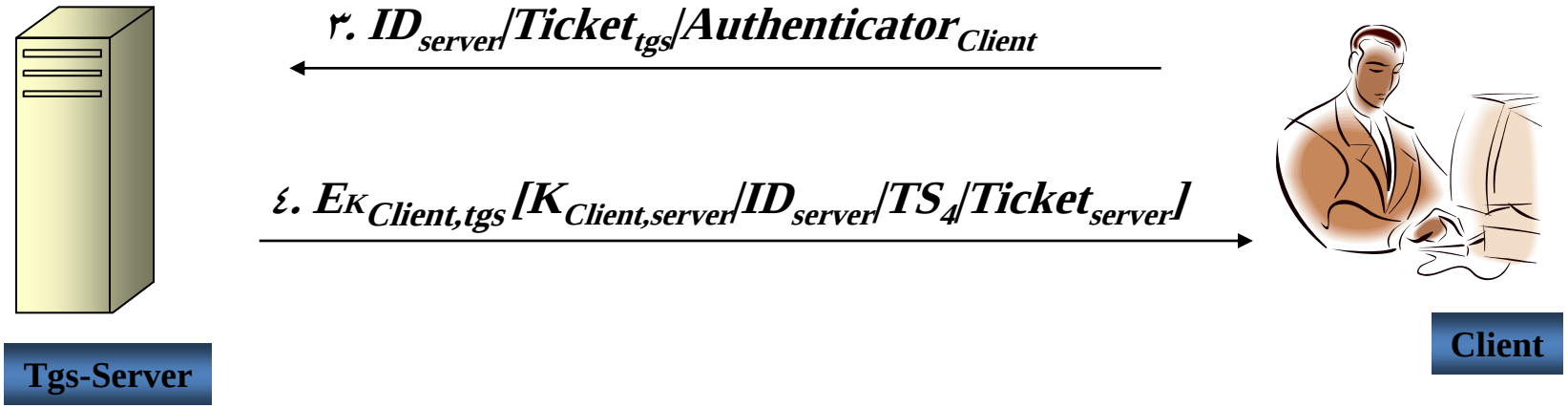
$$1.\ ID_{Client}/ID_{tgs}/TS_1$$

$$2.\ E_{KClient}[K_{Client,tgs}/ID_{tgs}/TS_2/Lifetime_2/Ticket_{tgs}]$$

Client

$$Ticket_{tgs}=E_{Ktgs}[K_{Client,tgs}/ID_{Client}/Addr_{Client}/ID_{tgs}/TS_2/Lifetime_2]$$

بدست آوردن بلیط "اعطاء خدمات"



$Ticket_{Server} =$

$E_{Kserver}[K_{Client,server}/ID_{Client}/Addr_{Client}/ID_{server}/TS_4/Lifetime_4]$

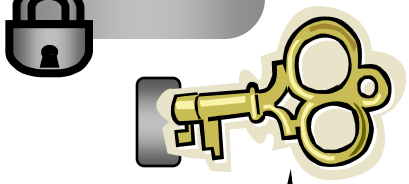
$Authenticator_{Client} =$

$E_{KClient,tgs}[ID_{Client}/Addr_{Client}/TS_3]$

بلیط کارگزار

$$Ticket_{Server} = E_{Kserver}[K_{Client,server}/ID_{Client}/Addr_{Client}/ID_{server}/TS_4/Lifetime_4]$$

تمامی با کلید کارگزار رمز شده اند

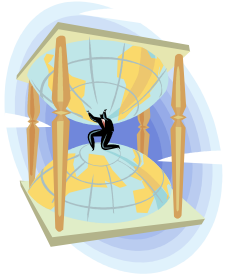


کلید جلسه بین کارفرما و کارگزار

شناسه کارفرما

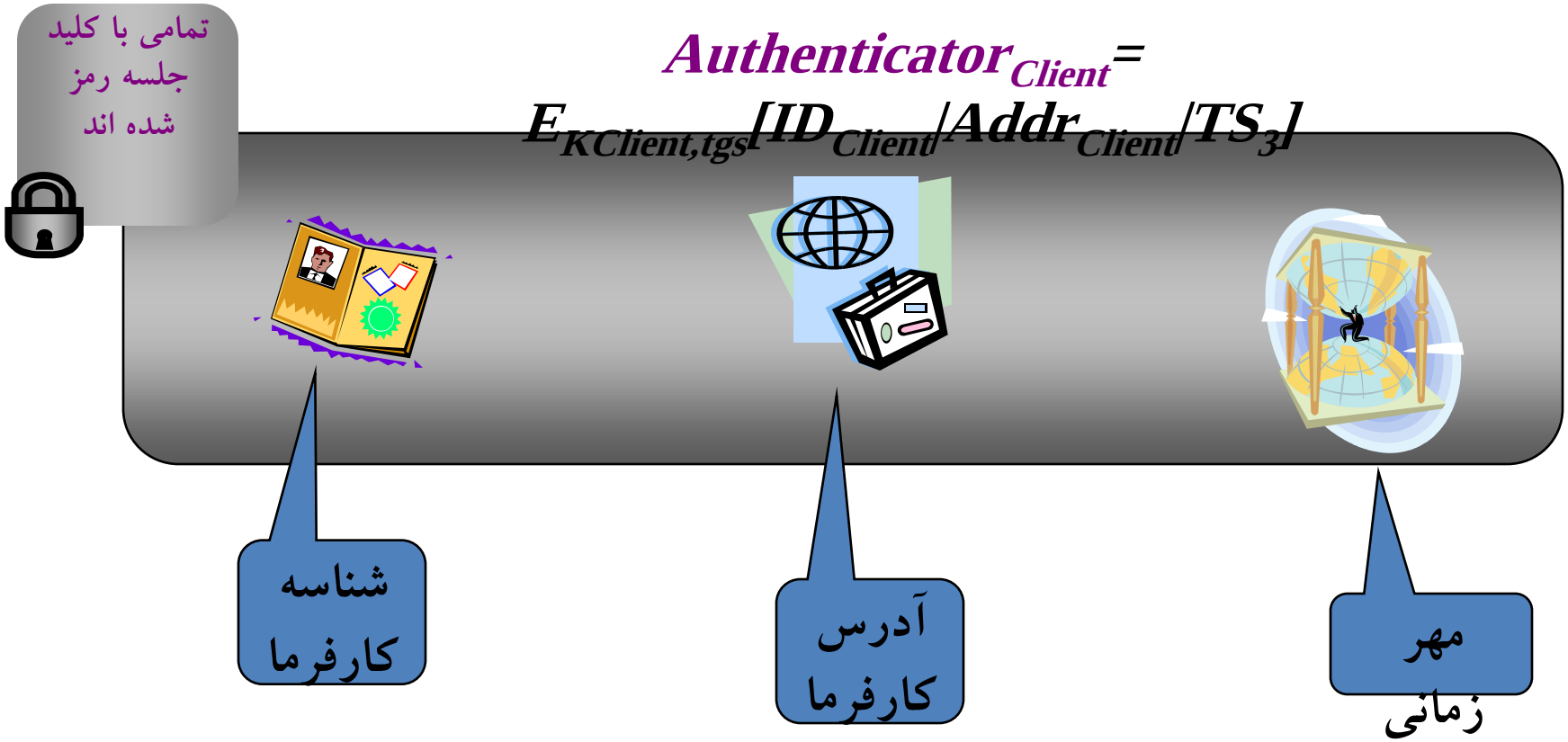
آدرس کارفرما

شناسه TGS



مهر زمانی و دوره اعتبار بلیط

اعتبار نامه کارفرما



نتایج این مرحله برای کارفرما

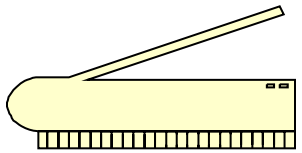
- جلوگیری از حمله تکرار با استفاده از یک اعتبار نامه (*Authenticator*) یکبار مصرف که عمر کوتاهی دارد.
- بدست آوردن کلید جلسه برای ارتباط با سرور

دستیابی به خدمات سرور

Client

5. $Ticket_{Server}|Authenticator_{Client}$

6. $E_{K_{Client,Server}}[TS_5+1]$



Server

نتایج این مرحله برای کارفرما

- احراز هویت کارگزار در گام ششم با برگرداندن پیغام رمزشده
- جلوگیری از بروز حمله تکرار

کربروس نسخه ٤: شمای کلی

(a) Authentication Service Exchange: to obtain ticket-granting ticket

(1) C → AS: $ID_c \| ID_{tgs} \| TS_1$

(2) AS → C: $\mathrm{E}_{K_c}\left[K_{c,tgs} \| ID_{tgs} \| TS_2 \| Lifetime_2 \| Ticket_{tgs}\right]$

$$Ticket_{tgs} = \mathrm{E}_{K_{tgs}}\left[K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2\right]$$

(b) Ticket-Granting Service Exchange: to obtain service-granting ticket

(3) C → TGS: $ID_v \| Ticket_{tgs} \| Authenticator_c$

(4) TGS → C: $\mathrm{E}_{K_{c,tgs}}\left[K_{c,v} \| ID_v \| TS_4 \| Ticket_v\right]$

$$Ticket_{tgs} = \mathrm{E}_{K_{tgs}}\left[K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2\right]$$

$$Ticket_v = \mathrm{E}_{K_v}\left[K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4\right]$$

$$Authenticator_c = \mathrm{E}_{K_{tgs}}\left[ID_C \| AD_C \| TS_3\right]$$

(c) Client/Server Authentication Exchange: to obtain service

(5) C → V: $Ticket_v \| Authenticator_c$

(6) V → C: $\mathrm{E}_{K_{c,v}}\left[TS_5 + 1\right]$ (for mutual authentication)

$$Ticket_v = \mathrm{E}_{K_v}\left[K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4\right]$$

$$Authenticator_c = \mathrm{E}_{K_{c,v}}\left[ID_C \| AD_C \| TS_5\right]$$

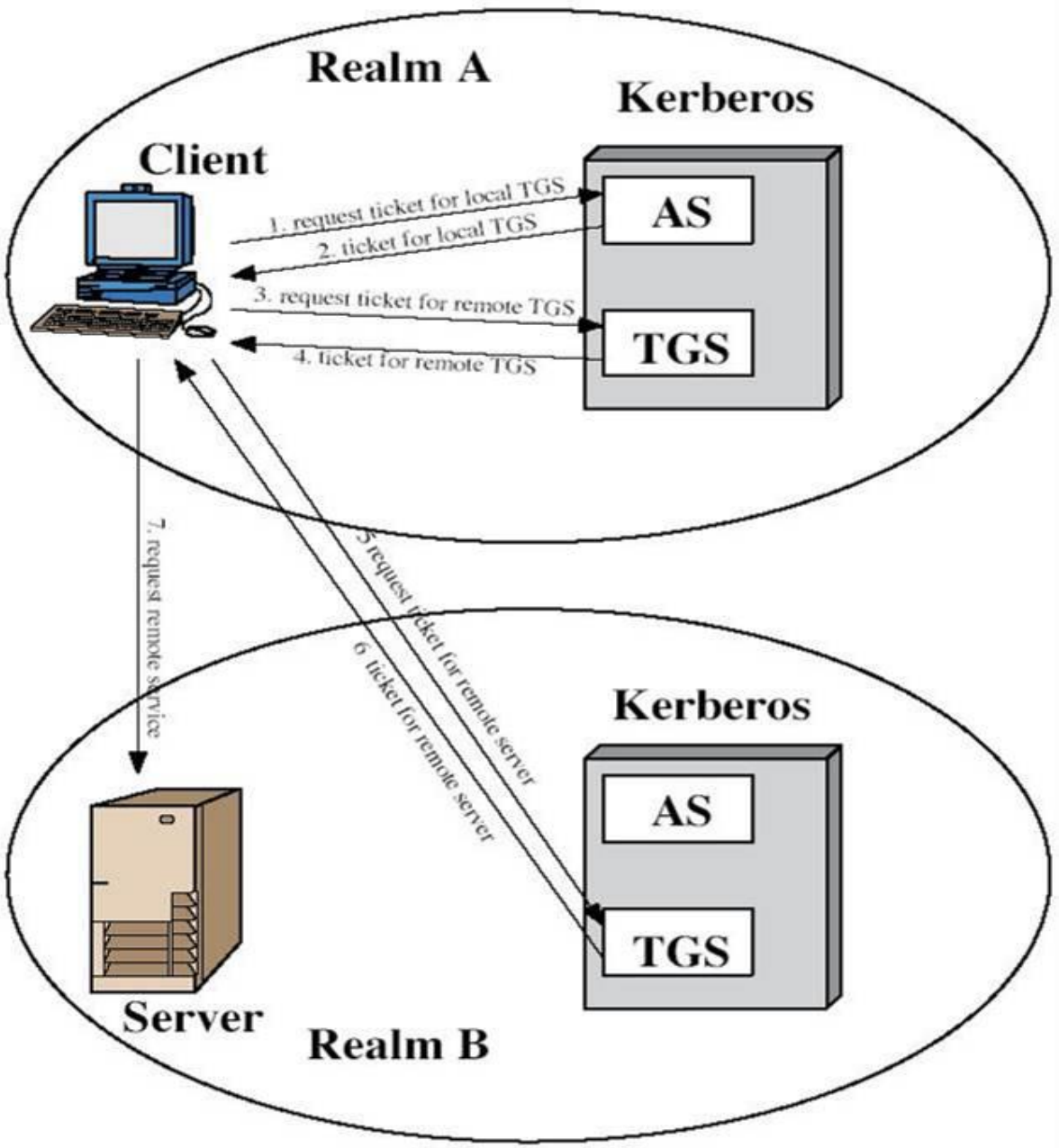
قلمرو کربروس

- قلمرو کربروس از بخشهای زیر تشکیل شده است:
  - کارگزار کربروس
  - کارفرمایان
  - کارگزاران کاربردی Application Servers
- کارگزار کربروس گذرواژه تمام کاربران را در پایگاه داده خود دارد.
- معمولاً هر قلمرو معادل یک حوزه مدیریتی می باشد.

هویت شناسی بین قلمرویی(InterRealm)

- امکان اینکه کاربران بتوانند از خدمات موجود در قلمروهای دیگر استفاده کنند.
  - کارگزاران کربروس هر قلمرو یک کلید مخفی با کارگزاران کربروس قلمرو همکار مقابل به اشتراک میگذارند.
  - وجود ***N*** قلمرو همکار نیازمند ***N(N-1)/2*** کلید مخفی است.
  - دو کارگزار کربروس همدیگر را ثبت نام مینمایند.

هویت شناسی بین قلمرویی



کربروس نسخه ۵

- مشخصات
  - در اواسط ۱۹۹۰ مطرح شد
  - نقص ها و کمبودهای نسخه قبلی را برطرف کرده است
  - به عنوان استاندارد اینترنتی ***RFC 1510*** در نظر گرفته شده است.
  - ویندوز ۲۰۰۰ از استاندارد اینترنتی کربروس نسخه ۵ بعنوان روش اصلی هویت شناسی کاربران استفاده می کند.

مشکلات Kerberos v4 و نحوه رفع آنها در نسخه ۵

- وابستگی به یک سیستم رمزنگاری خاص(DES)

  + در نسخه ۵ می توان از هر الگوریتم متقارن استفاده کرد

- وابستگی به IP

  + در نسخه ۵ می توان از هر نوع آدرس شبکه ای استفاده کرد

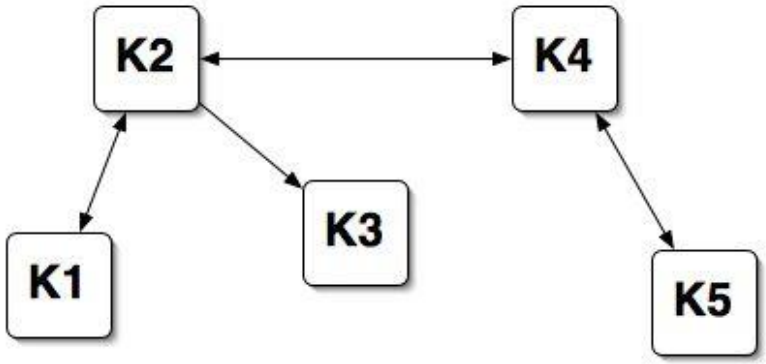
  – امکان استفاده از اعتبار کاربر متفاوت در دسترسی به یک سرور خاص

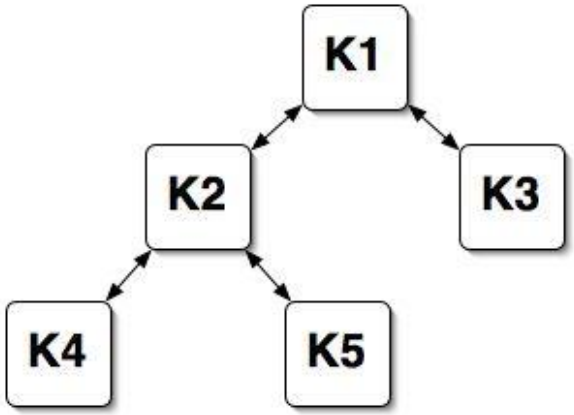
  + در نسخه ۵ اجازه داده می شود که سرویس از حساب کاربر متفاوت از کاربر login کرده انجام شود.

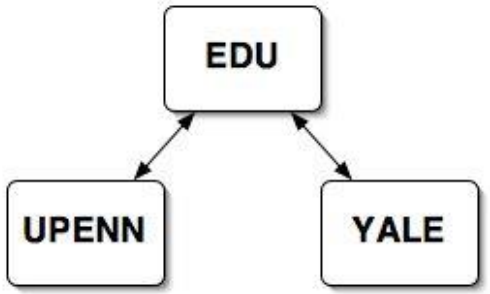
  – با افزایش تعداد قلمروها، تعداد کلیدها بصورت تصاعدی افزایش می یابد

  + در نسخه ۵ با استفاده از معماری سلسه مراتبی این مشکل حل شده است.

Hierarchy/Chain of Realms







کربروس نسخه ۵: شمای کلی

(a) Authentication Service Exchange: to obtain ticket-granting ticket

(1) $C \rightarrow AS$: $Options \| ID_c \| Realm_c \| ID_{tgs} \| Times \| Nonce_1$

(2) $AS \rightarrow C$: $Realm_c \| ID_c \| Ticket_{tgs} \| E_{Kc}[K_{c,tgs} \| Times \| Nonce_1 \| Realm_{tgs} \| ID_{tgs}]$

$Ticket_{tgs} = E_{K_{tgs}}[Flags \| K_{c,tgs} \| Realm_c \| ID_c \| AD_c \| Times]$

(b) Ticket-Granting Service Exchange: to obtain service-granting ticket

(3) $C \rightarrow TGS$: $Options \| ID_v \| Times \| \| Nonce_2 \| Ticket_{tgs} \| Authenticator_c$

(4) $TGS \rightarrow C$: $Realm_c \| ID_c \| Ticket_v \| E_{K_{c,tgs}}[K_{c,v} \| Times \| Nonce_2 \| Realm_v \| ID_v]$

$Ticket_{tgs} = E_{K_{tgs}}[Flags \| K_{c,tgs} \| Realm_c \| ID_c \| AD_c \| Times]$

$Ticket_v = E_{K_v}[Flags \| K_{c,v} \| Realm_c \| ID_c \| AD_c \| Times]$

$Authenticator_c = E_{K_{c,tgs}}[ID_c \| Realm_c \| TS_1]$

(c) Client/Server Authentication Exchange: to obtain service

(5) $C \rightarrow TGS$: $Options \| Ticket_v \| Authenticator_c$

(6) $TGS \rightarrow C$: $E_{K_{c,v}}[TS_2 \| Subkey \| Seq\#]$

$Ticket_v = E_{K_v}[Flags \| K_{c,v} \| Realm_c \| ID_c \| AD_c \| Times]$

$Authenticator_c = E_{K_{c,V}}[ID_c \| Realm_c \| TS_2 \| Subkey \| Seq\#]$