امنیت داده ها

فصل سیزدهم : مکانیزم های احراز هویت

دکتر یعقوب فرجامی

عضو هیات علمی دانشکده فنی قم

Authentication Mechanism

- ***Authentication*** : مکانیزمی که براساس آن پروسه ها هویت حقیقی و حقوقی کاربران خود را اثبات می کنند (احراز هویت)
- ***Authorization*** : مکانیزمی که براساس آن مشخص می شود پروسه ای که هویت واقعی آن احراز شده مجوز انجام چه عملیاتی را دارد (مجوزها- تعیین سطح دسترسی)
- ***Accounting*** : مکانیزمی که براساس آن مشخص می شود پروسه چه سهمی از منابع سیستمی و خدمات را می تواند استفاده کند (حسابرسی)

نکته : مهمترین و حساس ترین بخش کار احراز هویت است

پروتکل‌های هویت‌شناسی

- **هویت شناسی دو طرفه**
  - هر دو طرف ارتباط باید از هویت همدیگر مطلع شوند.

- **هویت شناسی یک طرفه**
  - لازم است تنها یک طرف ارتباط هویت خود را اثبات کند.
  - مورد استفاده : یک شخص یک پیام را در یک گروه عمومی منتشر می کند.

پروتکل‌های هویت‌شناسی

- **دو خطر اساسی تبادل امن کلیدهای جلسه را تهدید می کند**
  - **شنود**
  - **حملات تکرار یا** حملات بازگشتی(Replay Attacks)
  - انواع حملات تکرار (Replay Attacks)
- **Simple Replay** : گرفتن پیغام و ارسال آن بعد از مدتی
- **Logged Replay** : گرفتن پیغام و ارسال قبل از اتمام "پنجره زمانی"
- **Undetected Replay** : پیغام اصلی نمی رسد و فقط پیغام جعلی می رسد
- **Backward Replay** : پاسخ به پیغام ارسالی بجای گیرنده
  - وقتی اتفاق می افتد که از رمزنگاری مرسوم استفاده می کنیم و تفاوت بین پیغامهای ارسالی و دریافتی با مقایسه محتوای آنها ممکن نیست

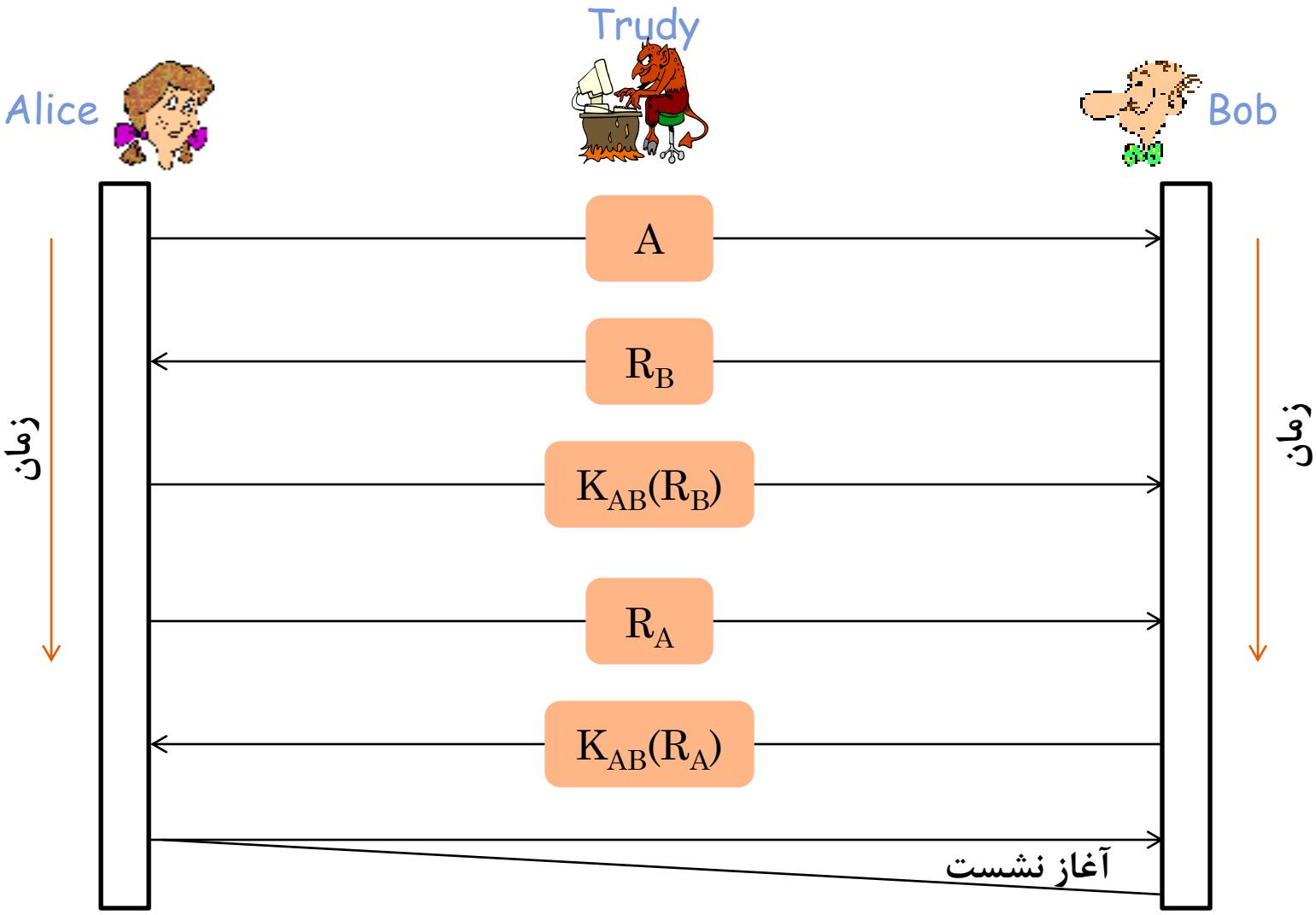
پروتکل‌های هویت‌شناسی

روشهای مقابله

- استفاده از اعداد متوالی(Sequence Number)
- استفاده از برچسب زمانی(TimeStamp) :

- Challenge/Response : قبل از ارسال هر پیغام، فرستنده یک Nonce ارسال می کند و انتظار دارد که گیرنده به آن پاسخ دهد.

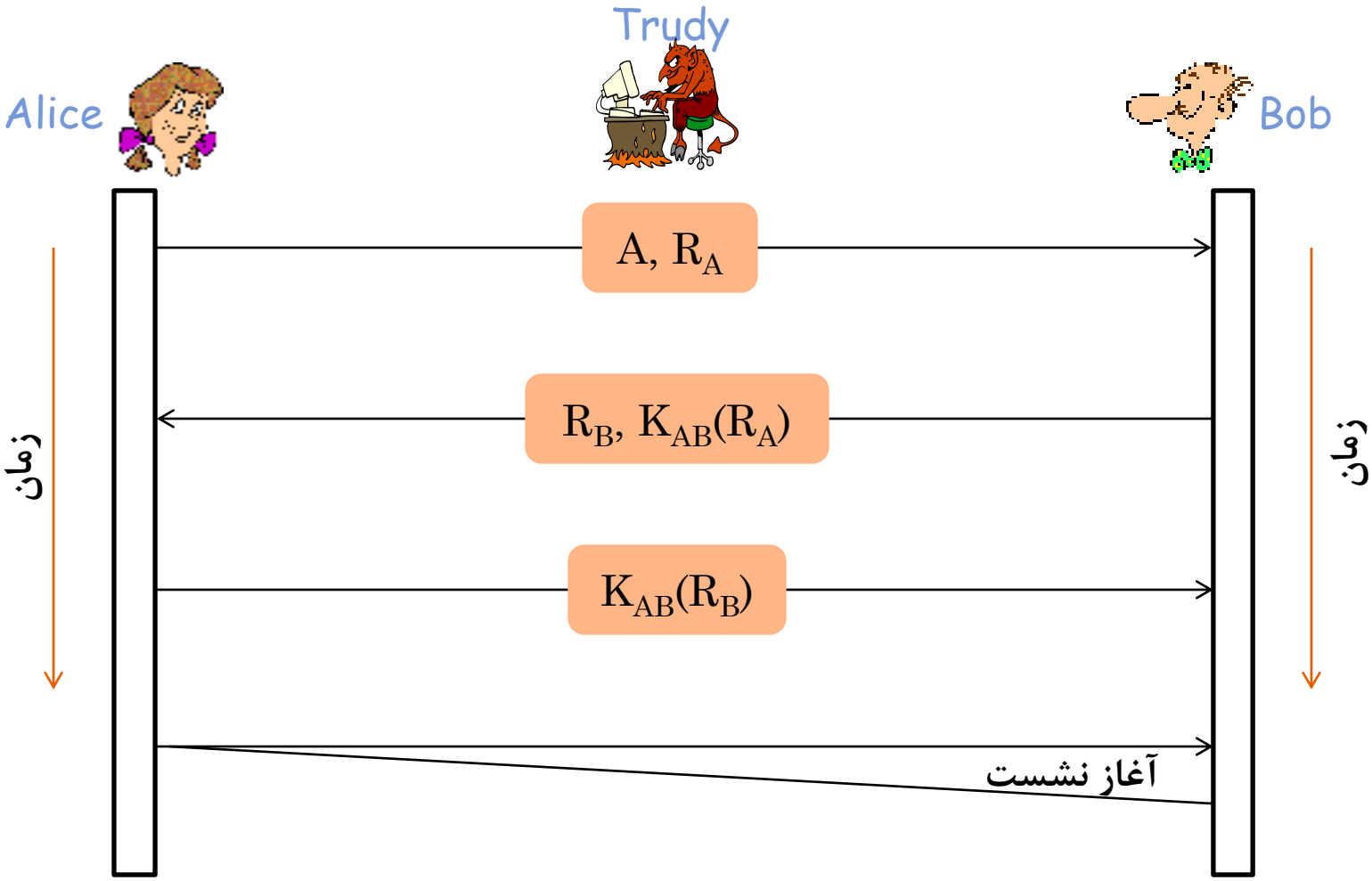
مکانیزم احراز هویت براساس «چالش و پاسخ»



مکانیزم احراز هویت براساس «چالش و پاسخ»

- آلیس و باب بر سر یک شاه کلید سری به توافق می رسند (K_{AB})(خارج از شبکه)
- یکی از طرفین(باب به عنوان سرور) برای شروع نشست یک رشته تصادفی (Nonce) تولید و برای طرف مقابل می فرستد (تا ادعای طرف مقابل مبنی بر این که آلیس است ثابت شود)
- طرف مقابل (آلیس به عنوان مشتری) تبدیلی روی رشته انجام داده و با شاه کلید رمز کرده، ارسال می کند
- این بار دو مرحله آخر باید بالعکس نیز انجام شود تا هویت طرف دیگر نیز محرز شود
- و بعد از پایان یافتن مبادله بسته های احراز هویت نشست شروع می شود

«چالش و پاسخ» خلاصه شده



«چالش و پاسخ» خلاصه شده

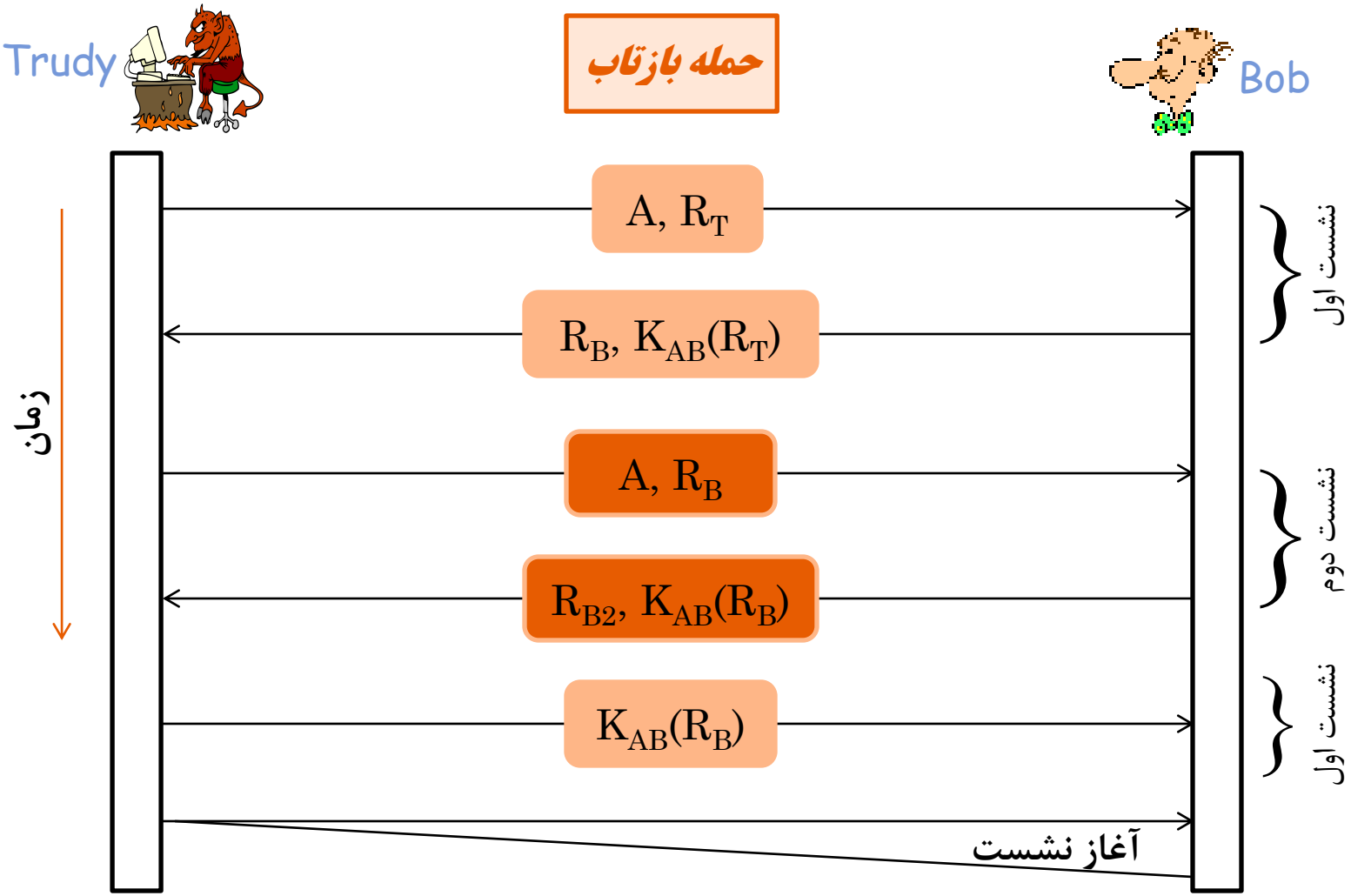
1. آلیس با ارسال شناسه کاربردی خود برای باب ، مشخص می کند که چه کسی است. شناسه کاربردی رمز نخواهد شد.
2. باب با دریافت شناسه کاربردی آلیس ، برای تشخیص هویت واقعی، یک رشته تصادفی بزرگ Rb تولید و با ارسال آن به آلیس ، او را به چالش می کشد.
3. آلیس رشته تصادفی Rbرا با کلید متقارن Kabرمزنگاری کرده به صورت Kab(Rb)و برای باب می فرستد .باب با کلید متقارن خود رشته Rbرمز را کرده و با رشته دریافتی از آلیس مقایسه می کند ، اگر رشته دریافتی با رشته رمزنگاری شده یکی بود در نتیجه هویت آلیس همانی هست که ادعا می کند.
4. آلیس با تولید رشته تصادفی Ra،باب را به چالش می کشد.
5. باب با رمزنگاری Ra،نتیجه را برای آلیس می فرستد و آلیس با مقایسه نتیجه رمزنگاری Raبا مقدار دریافتی ، از هویت باب مطمئن می شود.

اشکالات این پروتکل

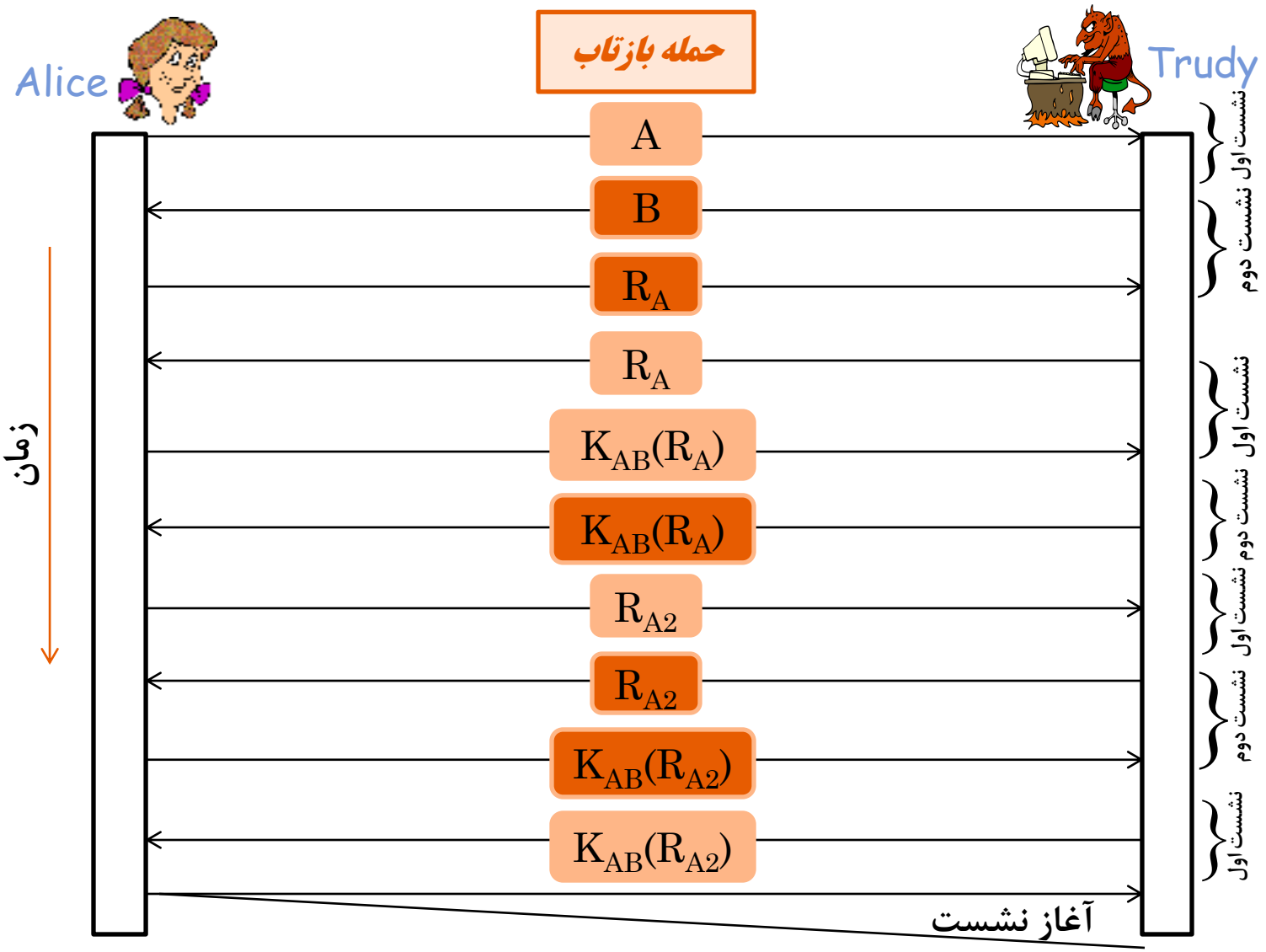
– مشتری می تواند سرویس دهنده را وادار کند که قبل از خودش هویت خود را اثبات کند .بدین ترتیب یک فریبکار قادر خواهد بود قبل از آنکه مدرکی در خصوص هویت خود ارائه بدهد اطلاعات باارزشی از ظرف مقابل کسب کند.

–گاهی سرویس دهنده ها مجبورند از نشست های موازی و همزمان پشتیبانی کند لذا باید مکانیزم احراز هویت به گونه ای طراحی شده باشد که نتوان اطلاعات بدست آمده از یک نشست را در نشست موازی دیگر به کار گرفت.

مشکلات پیش روی «چالش و پاسخ» خلاصه شده



مشکلات پیش روی «چالش و پاسخ»

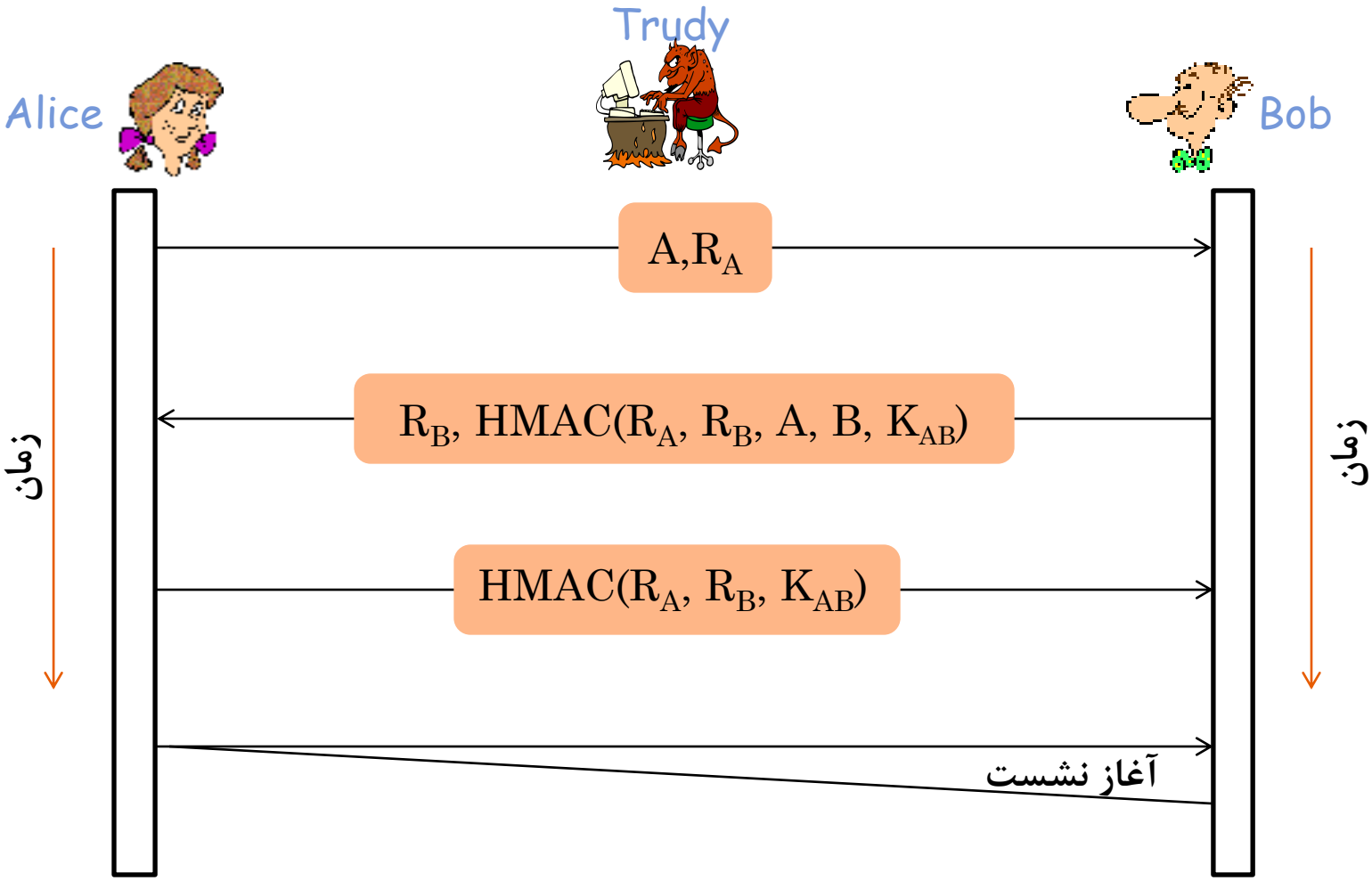


احراز هویت با استفاده از HMAC

- ***فرضیات :***
  - آلیس و باب فرض خواهند کرد طرف مقابل ترودی نیست
  - آلیس وباب قبلاً بر سر یک کلید سری به توافق رسیده اند (K_{AB})
- آلیس رشته چالش خود را ارسال می کند (R_A)
- باب در پاسخ، رشته چالش خود (R_B) و نیز چکیده ساختمان داده ای که با HMAC استخراج کرده می فرستد $HMAC(R_A, R_B, A, B, K_{AB})$
- آلیس هم در پاسخ به باب، رشته چالش خود وباب و کلید رمز را با HMAC استخراج و برای باب ارسال می کند $HMAC(R_A, R_B, K_{AB})$

نکته : HMAC صرفاً برای اثبات درستی ادعای طرفین است

احراز هویت با استفاده از HMAC



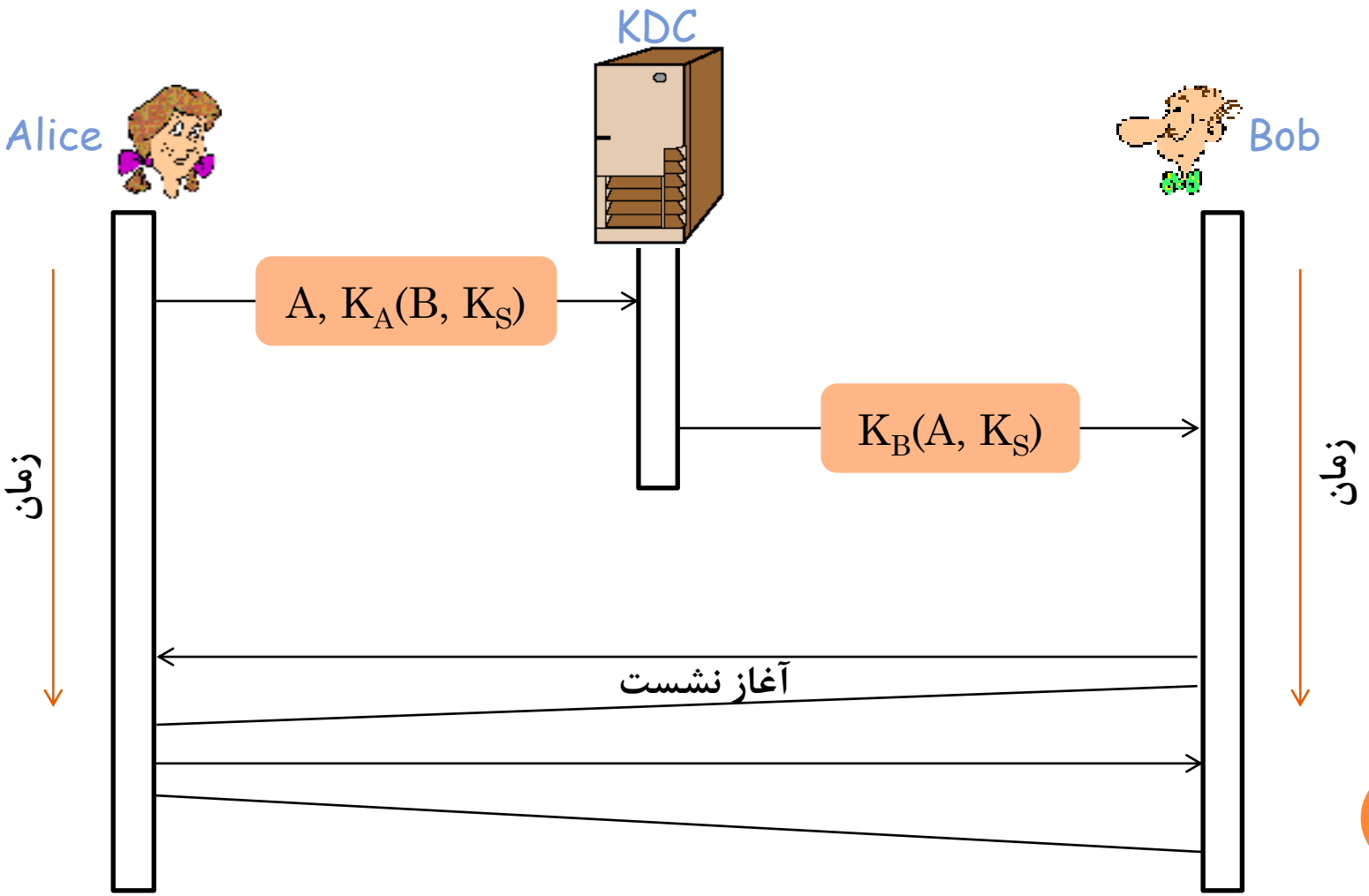
پروتکلهای احراز هویت با کمک مرکز توزیع کلید (KDC)

- استفاده از رمزنگاری مرسوم
  - سلسله مراتب دو لایه ای کلیدها (Session & Master keys)
  - مرکز توزیع کلید (KDC) مطمئن
    - هر شخص کلید اصلی خود را با KDC به اشتراک می گذارد
    - KDC کلید جلسه را تولید می کند
    - کلیدهای اصلی برای انتقال کلید جلسه به طرفین بکار می رود

احراز هویت با کمک مرکز توزیع کلید (KDC)

- مراجعه به مرکز توزیع کلید حداقل یکبار
- وجود یک مرکز برای ذخیره کلیدهای سری کاربران (KDC)
- احراز هویت افراد در یک نشست صرفاً با دخالت KDC
- آلیس یک کلید دلخواه (کلید نشست) انتخاب می کند و به همراه شناسه باب، آن را با کلید خودش رمز می کند
- آلیس ساختمان داده بالا را با شناسه خودش به KDC ارسال می کند
- KDC از روی شناسه دریافتی کلید آلیس را انتخاب کرده و ساختمان داده را رمزگشایی می کند
- KDC با کمک کلید باب ساختمان داده جدیدی که شامل شناسه آلیس و کلید نشست است را رمز کرده به باب می فرستد و کارش پایان می یابد
- حال آلیس و باب با اطمینان و با کمک کلید توافقی، نشست را آغاز می کنند

احراز هویت با کمک مرکز توزیع کلید (KDC)



احراز هویت با کمک مرکز توزیع کلید (KDC)

- **مشکلات** :
- خطر *حمله تکرار* وجود دارد، زمانی که ترودی پیام KDC به باب را تکرار کند
- **راه حل** :
- *استفاده از مهر زمان* (Time stamp)
- هر چند مهر زمان خطر حمله تکرار را کاهش می دهد ولی کامل از بین نمی برد
- *استفاده از رشته تصادفی* (Nonce)
- که مشکل فضای ذخیره سازی زیاد را به دنبال دارد
- *استفاده ترکیبی از مهر زمان و رشته تصادفی*

احراز هویت با کمک مرکز توزیع کلید (KDC)

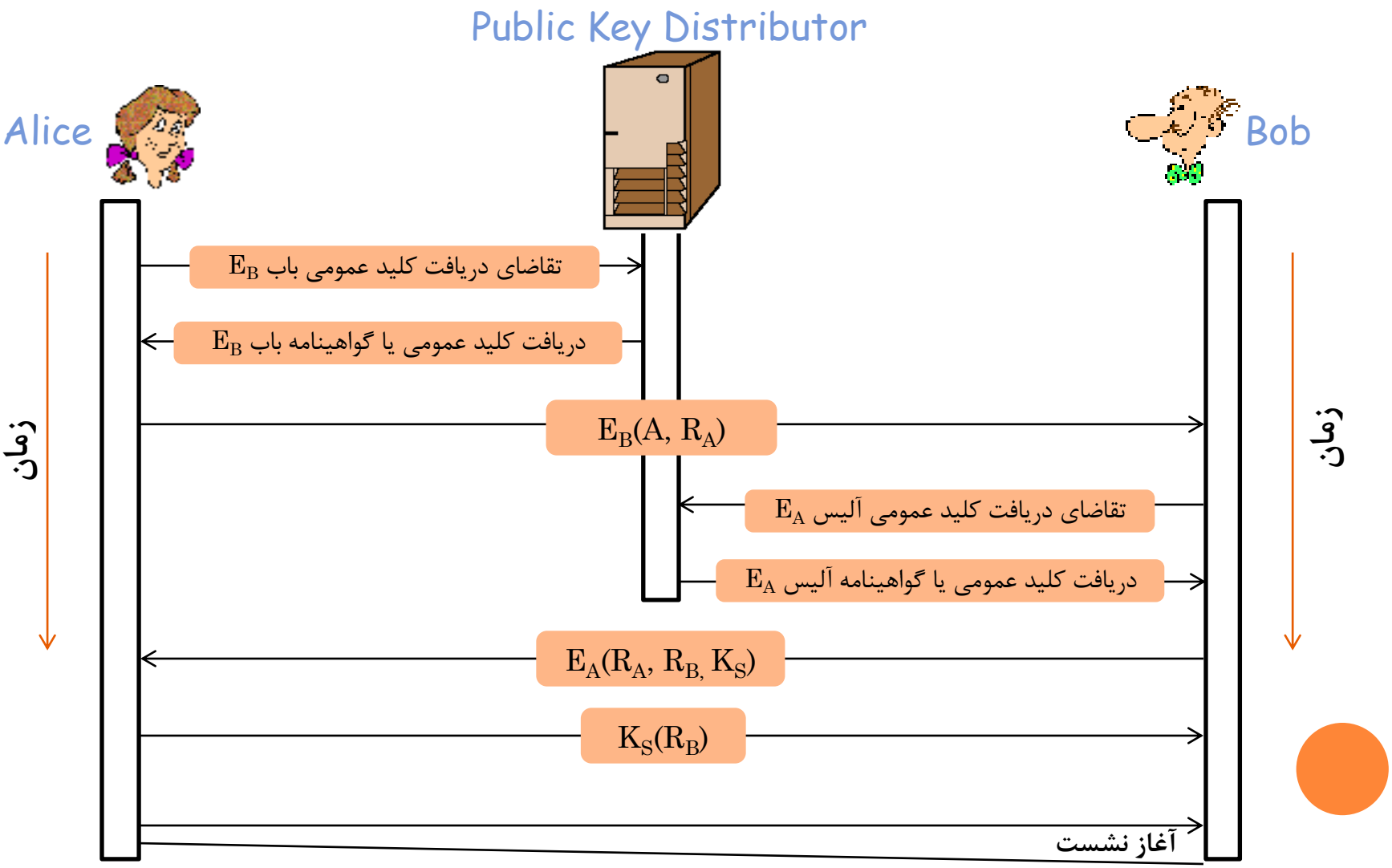
- استفاده از رمزنگاری مرسوم

A → KDC $ID_A||ID_B||N1$

KDC → A $E_{K_A}[K_S||ID_B||N1||E_{K_B}[K_S||ID_A]]$

A → B $E_{K_B}[K_S,ID_A]||E_{K_S}[M]$

احراز هویت با رمزنگاری کلید عمومی



پروتکلهای هویت‌شناسی

- استفاده از کلید عمومی
  - هدف : محرمانگی

A → B $E_{Kub}[K_S] \,||\, E_{K_S}[M]$

  - هدف : احراز هویت

A → B $M \,||\, E_{KR_A}[H(M)]$

  - احراز هویت ، بدون اطلاع طرفین از کلید عمومی یکدیگر

A → B $M \,||\, E_{K_{RA}}[H(M)] \,||\, E_{KR_{AS}}[T \,||\, ID_A \,||\, K_{UA}]$

پروتکلهای هویت‌شناسی

کلید عمومی و برچسب زمانی

1. $A \rightarrow AS : ID_A \mid ID_B$

2. $AS \rightarrow A : E_{KRas}[ID_A \mid KU_a \mid T] \mid E_{KRas}[ID_B \mid KU_b \mid T]$

3. $A \rightarrow B : E_{KRas}[ID_A \mid KU_a \mid T] \mid E_{KRas}[ID_B \mid KU_b \mid T] \mid E_{KUb}[E_{KRa}[K_s \mid T]]$

مشکل : سنکرون بودن زمان سیستم های طرفین

احراز هویت با رمزنگاری کلید عمومی

- استفاده از رمزنگاری عمومی

  - طرفین نیاز به دانستن کلید عمومی فعلی هم ندارند
  - کارگزار شناسایی(AS) علاوه بر توزیع کلید جلسه، وظیفه ایجاد گواهی کلید عمومی را بر عهده دارد
  - مانند رمزنگاری مرسوم، می توان از برچسب زمانی یا nonce استفاده کرد

پروتکلهای هویت‌شناسی

- کلید عمومی و nonce

1. $A \rightarrow KDC \quad : ID_A \mid ID_B$

2. $KDC \rightarrow A \quad : E_{KRauth}[ID_b \mid\mid KU_b]$

3. $A \rightarrow B \quad : E_{KUb}[N_a \mid\mid ID_A]$

4. $B \rightarrow KDC \quad : ID_B \mid\mid ID_A \mid\mid E_{KUauth}[N_a]$

5. $KDC \rightarrow B \quad :$

$$E_{KRauth}[ID_A \mid\mid KU_a] \mid\mid E_{KUb}[E_{KRauth}[N_a \mid\mid ID_A \mid\mid ID_B]]$$

6. $B \rightarrow A \quad : E_{KUa}[E_{KRauth}[N_a \mid\mid Id_A \mid\mid ID_B \mid\mid N_b]]$

7. $A \rightarrow B \quad : E_{Ks}[N_b]$

احراز هویت با رمزنگاری کلید عمومی

- **فرض**
- موسسه ای برای کاربران خود گواهینامه x.509 صادر و PKI خود را راه اندازی کرده است
- **روال کار**
- در اولین گام آلیس از سرور توزیع کننده کلید عمومی، تقاضای دریافت کلید عمومی باب را دارد
- اگر سیستم بر مبنای PKI باشد می توان گواهینامه باب را اعتبارسنجی و کلید عمومی او را استخراج نمود

توجه : تنها کاری که ترودی می تواند انجام دهد استراق سمع پیام سوم است و می تواند ان را تکرار کند، اما از جایی که کلید آلیس را ندارد نمی تواند پاسخ باب را رمزگشایی کند و نشست به مرحله آخر نمی رسد ☺

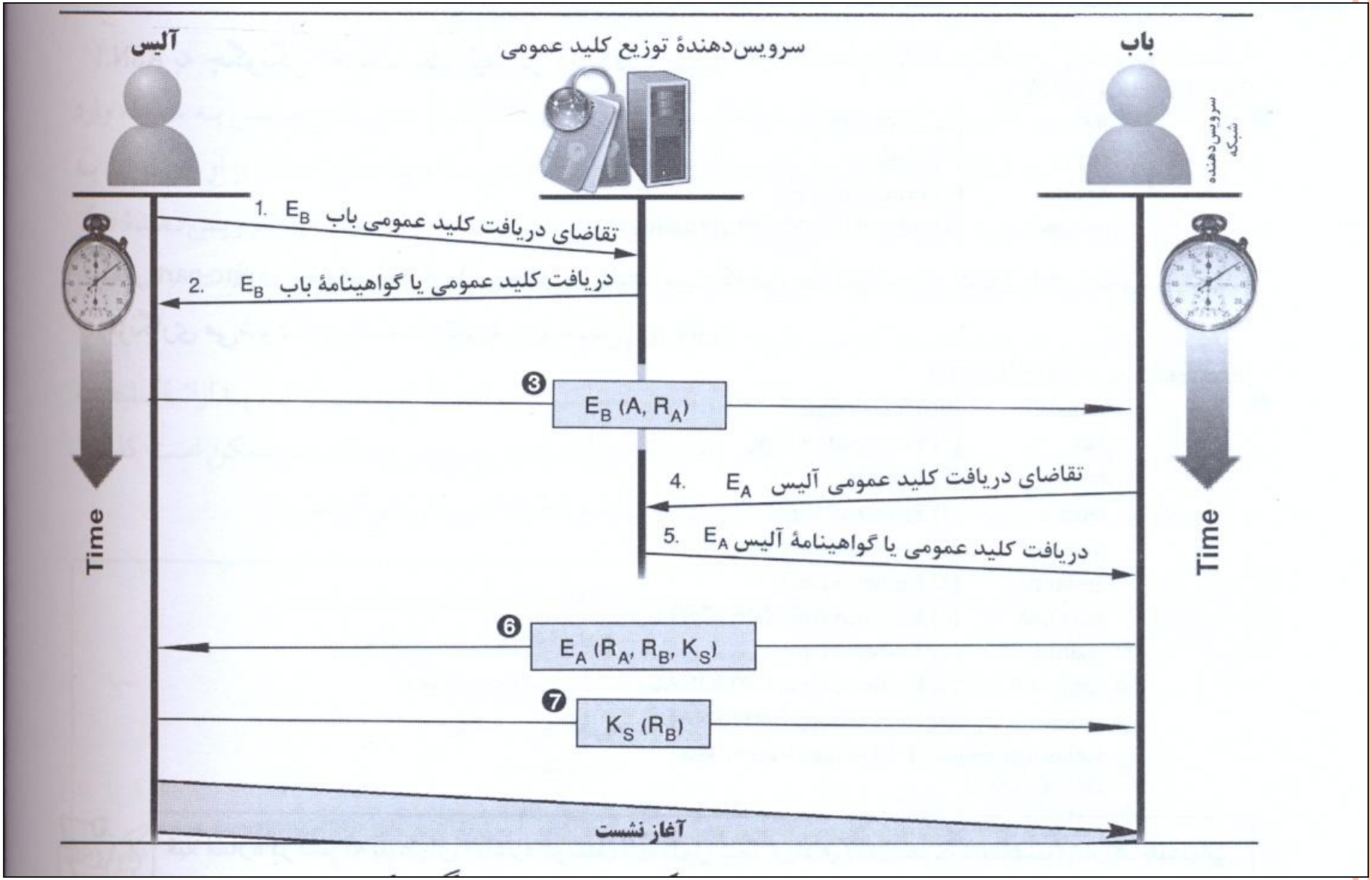
احراز هویت با رمزنگاری کلید عمومی

- در اولین گام، آلیس باید از سرویس دهنده ی توزیع کلید عمومی، تقاضای دریافت کلید باب را بدهد. هر گاه سیستم مبتنی بر PKI بنا نهاده شده باشد او می تواند با دریافت گواهینامه ی دیجتالی صادره برای باب، آن را اعتبار سنجی کرده و سپس کلید عمومی باب را از درون گواهینامه بدست آورد.
- در این مرحله نیز فرض شده که آلیس به روشی مطمئن، کلید عمومی مندرج در گواهینامه ی دیجتالی باب را بدست آورده و قادر است به کمک آن اطلاعاتی را رمز کند و برای باب بفرستد.
- در پیام سوم، آلیس شناسه ی کاربردی خود (A) و عدد تصادفی Ra را در یک ساختمان داده ی مشخص قرار داده و آن را به کمک کلید عمومی باب (Eb) رمزنگاری و ارسال می کند. بدیهی است که هیچکسی به جز خود باب قادر به رمزگشایی این پیام نیست.
- باب پس از رمزگشایی پیام سوم با کلید خصوصی خود، شناسه ی آلیس و رشته ی چالش او را استخراج می کند و چون برای پاسخ به آلیس، کلید عمومی او را نیاز دارد، به همین دلیل او نیز در پیام چهارم از سرویس دهنده ی توزیع کلید، کلید عمومی یا گواهینامه ی آلیس را تقاضا می کند.

- پس از دریافت کلید عمومی یا گواهینامه ی آلیس، شرایط برای ارسال اطلاعات رمزنگاری شده مهیاست.
- در این پیام، باب سه قسمت:Ra (رشته چالش ارسالی توسط آلیس)، Rb (رشته چالش خودش) و Ks (کلید نشست برای استفاده از آن در رمزنگاری متقارن داده ها) را در یک ساختمان داده مشخص قرار داده و حاصل را به کمک کلید عمومی آلیس، رمزنگاری کرده و آن را برای او پس می فرستد. طبیعتا تنها کسی که قادر به رمزگشایی و بازخوانی این قسمت ها خواهد بود، آلیس است.

- آلیس پس از رمزگشایی داده ها، ابتدا با بررسی Ra و مقایسه ی آن با رشته ی چالش ارسالی به این نتیجه می رسد که این پاسخ دقیقا با درخواست او تطابق دارد یا خیر ؟ بدین ترتیب تازه بودن پیام اثبات و از حمله تکرار جلوگیری می شود. حال باید با ارسال Rb که با کلید نشست Ks رمزنگاری شده، به باب ثابت کند که او نیز یک کاربر فعال و واقعی است وپیام سوم، پیامی تکراری نبوده است. بدین ترتیب نشست بین باب و آلیس با کلید نشست Ks آغاز خواهد شد.

احراز هویت با رمزنگاری کلید عمومی



احراز هویت با پروتکل نیدهام–شرودر

- مبتنی بر «چالش و پاسخ» و مرکز توزیع کلید (KDC)
- ارسال شناسه آلیس و باب و رشته تصادفی آلیس به KDC
- ارسال پاسخ به شکل یک ساختمان داده از KDC به آلیس و پایان کار KDC

$K_A(R_A, B, K_S, K_B(A, K_S))$

- نکته : $K_B(A, K_S)$ بلیطی از طرف KDC برای آلیس است که باید تسلیم باب کند
- با استخراج بلیط و کلید نشست، آلیس بلیط و رشته چالشی جدید خود را که با کلید نشست رمز شده را برای باب می فرستد
- برای اثبات هویت باب به آلیس رشته چالش آلیس را تغییر داده با کلید نشست رمز می کند و به همراه رشته چالش خود ارسال می کند
- آلیس هم برای اینکه به باب ثابت کند که پیام های قبلی او قدیمی نیستند رشته چالشی باب را با کلید نشست رمز کرده و ارسال می کند

احراز هویت با پروتکل نیدهام–شرودر

1. آلیس ابتدا یک کلید نشست دلخواه و تصادفی به نام Ks برای خودش انتخاب و آن را همراه شناسه کاربری باب B در یک ساختمان داده مشخص قرار داده وپس از رمزنگاری آن را به کمک کلید سری خودش Ka ، نتیجه بدست آمده را به همراه شناسه کاربری خود برای KDC می فرستد . هیچکس در جهان به جز آلیس و KDC قادر به رمز گشایی Ka(B,Ks) و بهره برداری از محتویات آن نخواهد بود . (حتی باب)

2. ۲. مرکز KDC پس از رمزگشایی قسمت دوم با کلید آلیس با چه کسی کار دارد و کلید نشست پیشنهادی او چیست . لذا شناسه کاربردی آلیس و کلید نشست پیشنهادی او را در یک ساختمان داده قرار داده و آن را با کلید سری باب رمز کرده و برای باب می فرستد . لذا این اطمینان وجود خواهد داشت که هیچکس در جهان به چز باب قادر به رمز گشایی و استخراج Ks از درون پیام نخواهد بود . اگر باب توانست پیام دریافتی را به درستی از رمز خارج کند مطمئن خواهد شد که پیام قطعا از طرف KDC آمده است چرا که هیچکس در جهان به جز KDC کلید سری او را نمی داند.

- -پس از رمز گشایی پیلم دوم و استخراج کلید نشست Ks نقش مرکز KDC نیز به پایان می رسد زیرا پس از این مراحله باب و آلیس داده ها و پیامهای خود را با کلید Ks رمز نگاری و بین یکدیگر رد و بدل خواهند کرد.
- -مشکل این روش

- مشکل حمله تکرار ، توانای رفع این مشکل با رویکرد مهر زمان و یا رویکرد رشته تصادفی ، بهترین روش تلفیق دو رو یکرد با هم است.

پروتکلهای هویت‌شناسی

- پروتکل Needham-Schroeder

1. A→KDC: $ID_A | ID_B | N_1$
2. KDC→A: $E_{Ka}[K_s | ID_B | N_1 | E_{Kb}[K_s | ID_A]]$
3. A→B: $E_{Kb}[K_s | ID_A]$
4. B→A: $E_{Ks}[N_2]$
5. A→B: $E_{Ks}[f(N_2)]$

پروتکلهای هویت‌شناسی

- پروتکل فوق نسبت به Replay Attack آسیب پذیر است
  - ممکن است کلید جلسه قبلی لو رفته باشد و بتوان جلسه جدیدی تشکیل داد.
- راه حل : اضافه کردن برچسب زمانی

1. $A \rightarrow KDC: \ ID_A \mid ID_B$
2. $KDC \rightarrow A: \ E_{Ka}[K_s \mid ID_B \mid T \mid E_{Kb}[K_s \mid ID_A / T]]$
3. $A \rightarrow B: \ E_{Kb}[K_s \mid ID_A / T]$
4. $B \rightarrow A: \ E_{Ks}[N_1]$
5. $A \rightarrow B: \ E_{Ks}[f(N_1)]$

پروتکلهای هویت‌شناسی

- حمله Suppress-Replay و مقابله با آن
  - پروتکل فوق نسبت به حمله Suppress_Replay آسیب پذیر است.
    - این حمله از سنکرون نبودن clock های فرستنده و گیرنده ناشی می شود. وقتی clock فرستنده جلوتر از clock گیرنده باشد.
  - روشهای مقابله :
    - چک کردن متناوب با زمان KDC
    - توافق از طریق nonce

پروتکلهای هویت‌شناسی

پروتکل بهبودیافته(جهت مقابله با حمله Suppress-Attack)

1. A→B: $ID_A \mid N_a$
2. B→KDC: $ID_B \mid N_b \mid E_{Kb}[ID_A / N_a / T_b]$
3. KDC→A: $E_{Ka}[ID_B \mid N_a \mid K_s \mid T_b] \mid E_{Kb}[ID_A \mid K_s \mid T_b] \mid N_b$
4. A→B: $E_{Kb}[ID_A \mid K_s \mid T_b] \mid E_{Ks}[N_b]$

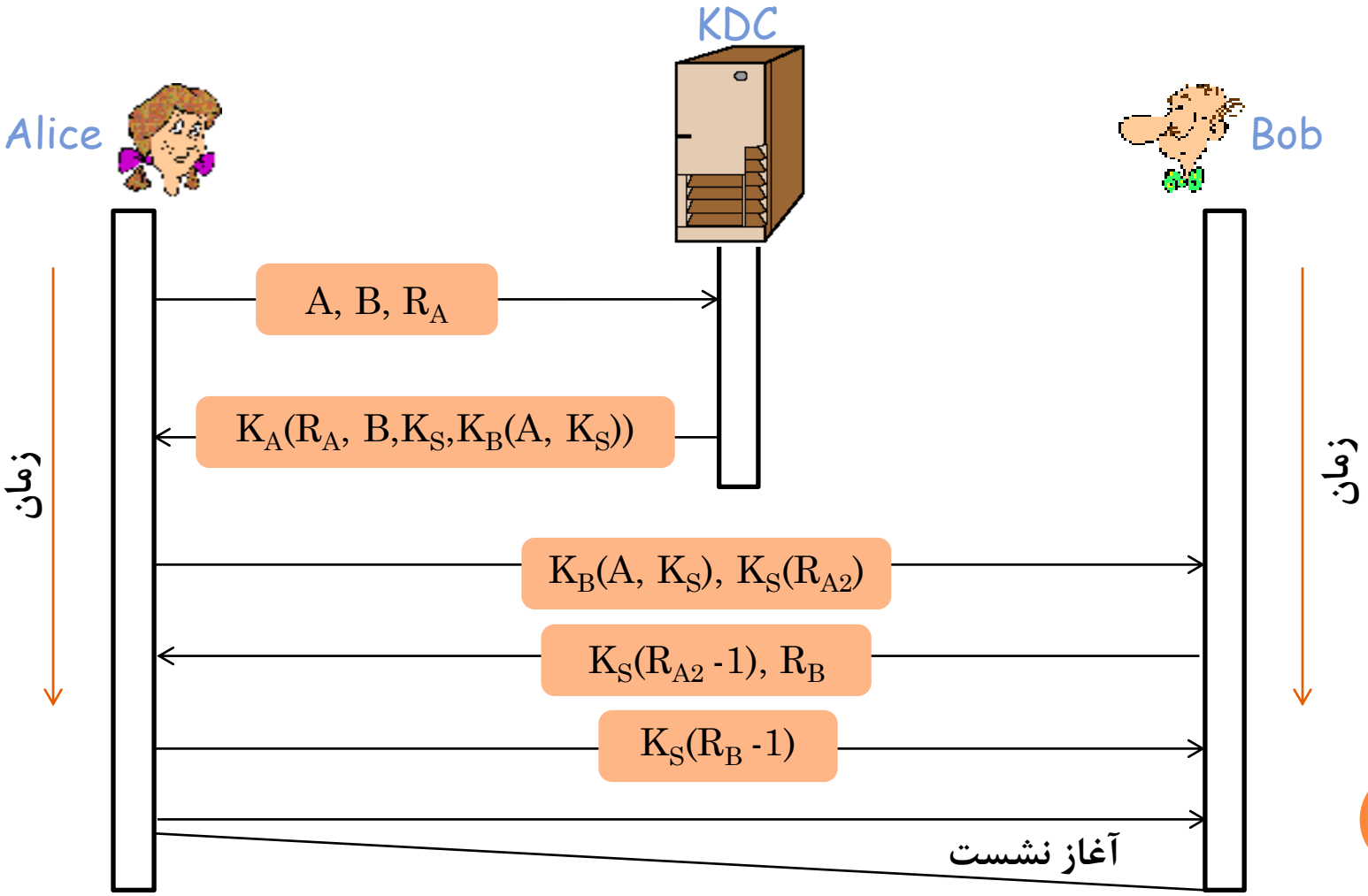
پروتکلهای هویت‌شناسی

- ۱ آلیس در اولین گام ، شناسه کاربردی خودش A ، شناسه کاربردی باب (سرویس دهنده B) و رشته تصادفی Ra را برای مرکز توزیع کلید می فرستد.
- ۲ مر کز توزیع کلید ساختمان داده ای با پنج قسمت زیر تولید و با کلید سری آلیس رمز کرده و برای او پس می فرستد :
- Ra رشته تصادفی آلیس.
- B شناسه کاربری باب.
- Ks یک کلید تصادفی که آلیس و باب در خلال نشست باید از آن استفاده کنند.
- Kb(A,Ks) یک قسمت حاصل رمز نگاری شناسه کاربری آلیس و کلید نشست Ks که با کلید سری باب رمز شده است و طبعا آلیس و هیچکس دیگر ، قادر به بهره برداری از درون آن نیست . آلیس باید این قسمت را پس از استخراج ، بدون هیچ تغییری برای باب بفرستد.
- توجه : به قسمت Kb(A,Ks) که آلیس باید تسلیم باب کند اصطلاحا «بلیط» گفته می شود.
- ۳ حال آلیس پس از رمز گشایی قسمت های پیامی که KDC فرستاده ، می تواند مولفه «بلیط » و کلید نشست Ks را استخراج کند. لذا در پیام سوم با انتخاب یک رشته چالش بزرگ دیگر Ra2 آن را با کلید نشست رمز کرده و نتیجه را همراه بلیط برای باب می فرستد .
- باب تنها کسی است که می تواند بلیط را از رمز خارج کند . با رمز گشایی بلیط ، باب به هویت آلیس پی می برد و کلید نشست را به دست می آورد . با استخراج کلید نشست باب می تواند مقدار Ra2 را رمز گشایی کند. تا اینجا هویت آلیس برای باب اثبات شده است زیرا هیچکس در جهان به جز KDC کلید باب را نمی داند و قطعا این مرکز KDC بوده که بلیط را به نام آلیس صادر کرده است.

4 برای انکه باب نیز هویت خود را به آلیس اثبات کند پس از رمز گشایی Ra2 با کلید نشست ، یک واحد از آن کسر و نتیجه را بار دیگر با کلید نشست رمز کرده و به همراه رشته چالش خود Rb برای آلیس پس می فرستد.

5 آلیس پس از دریافت قسمت های پیام چهارم ، ابتدا با کلید نشست مقدار Ra2 -1 را رمز گشایی کرده ، یک واحد به آن اضافه خواهد کرد ، و با مقدار ارسالی مقایسه می کند تا مطمئن شود پیام دریافتی دقیقا در پاسخ به تقاضای کنونی بوده وتکراری نباشد یا جعلی نیست.. حال آلیس مقدار Rb را پس از کسر یک واحد ، براب باب می فرستد تا حضور فعال او برای باب اثبات شود و باب نیز اطمینان حاصل کند که پیام سوم متعلق به زمان های قبل و تکراری نیست.

احراز هویت با پروتکل نیدهام–شرودر



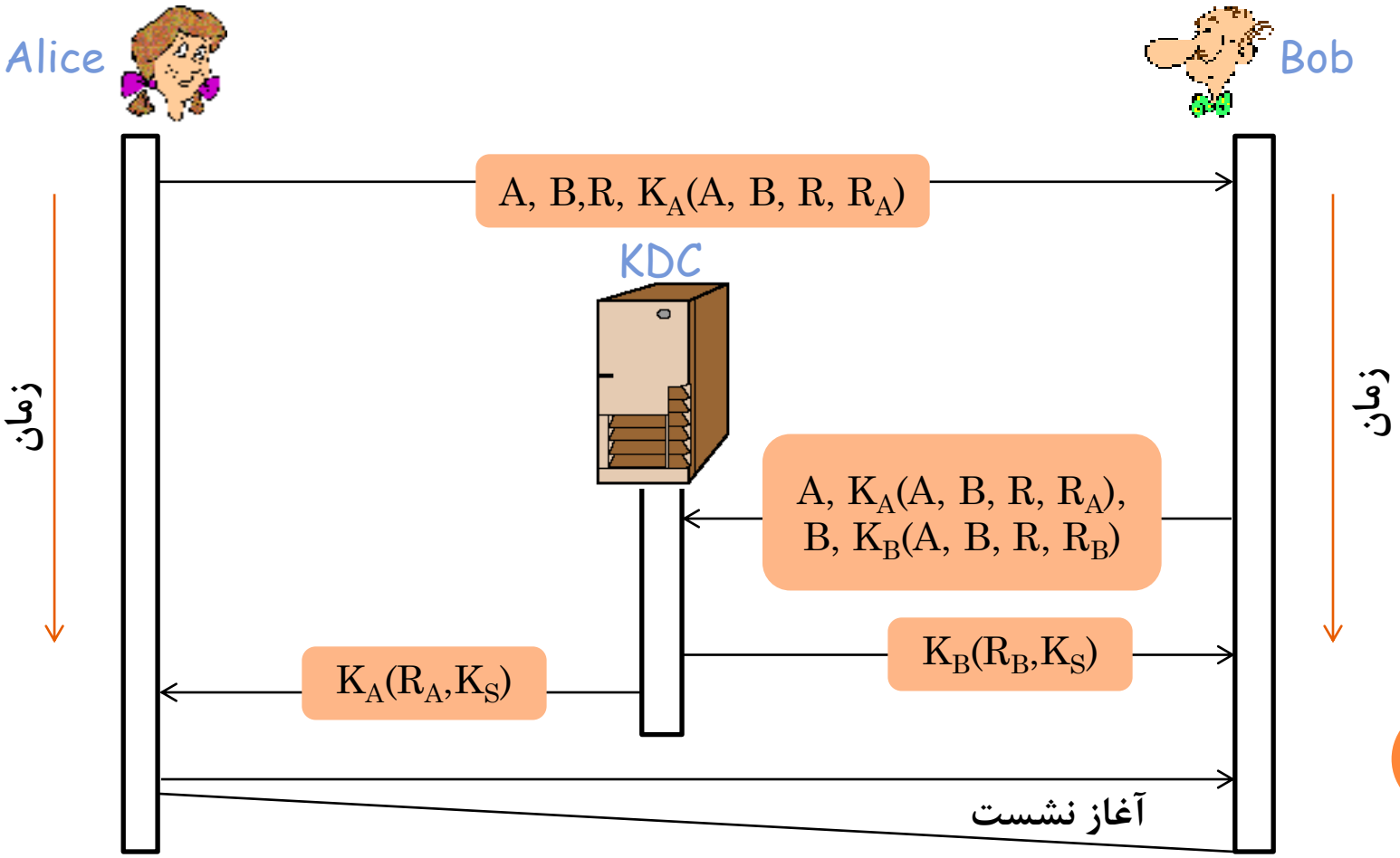
احراز هویت با پروتکل آتوی–ریس

- چون تنها KDC کلید سری افراد را می داند، آلیس و باب با دیدن رشته های چالش خود مطمئن می شوند که پیام تکراری نیست
- اگر ترودی در پیام اول تغییراتی روی پارامترهای آشکار انجام دهد توسط KDC کشف می شود

احراز هویت با پروتکل آتوی–ریس

- آلیس کار خوئد را با تولید دو عدد تصادفی بزرگ آغاز می کند : Rکه به عنوان شناسه کاربردی مشترک بکار می رود و Raکه همان رشته چالش آلیس به شمار می آید . در اولین پیام آلیس شناسه کاربردی خود A، شناسه کاربردی باب Bو شناسه تصادفی ومشترک Rرا به صورت آشکار به همراه یک قسمت رمز نگاری شده چهارم برای باب می فرستد. قسمت چهارم در حقیقت حاصل رمز نگاری چهار آیتم a، b، rو Raٔ است که با کلید متقارن آلیس رمز شده است . از بین این چهار قسمت فقط raاست که از چشم ترودی مخفی می ماند.
- با دریافت پیام اول ، باب نمی تواند درباره صحت این پیام تصمیمی بگیرد زیرا کلید آلیس را ندارد. لذا بلافاصله عدد تصادفی Rbرا را به سه قسمت A، Bو rافزوده ونتیجه را با کلید سری خود ش رمز نگاری کرده و به همراه پیام دریافتی از آلیس ، به مرکز KDCمی فرستد.
- مرکز KDCکلید همه را در اختیار دارد لذا اولین قسمت را با کلید الیس و دومین قسمت را با کلید باب از رمز خارج و در اولین گام مقدار rهر دو قسمت را استخراج و با هم مقایسه می کند تا ثابت شود این دو پیام در پاسخ یکدیگر تولید شده اند . اگر هر دو rیکی باشند در نتیجه KDCمتقاعد می شود که پیام ارسالی از باب در پاسخ به تقاضای کنونی آلیس صادر شده است . KDCبرای الیس و باب یک کلید نشست Ksتولید می کند و آنرا همرا رشته چالش آنها ، پس از رمز نگاری با کلید متناظر ، برای ایشان می فرستد.
- پس از دریافت Ka(Ra,ks)آلیس آنرا از رمز خارج کرده و Raرا با رشته چالشارسالی خود مقایسه می کند ، همین کار توسط باب بر روی Kb(Rb, Ks)انجام می شود . هرگاه دو طرف رشته چالش خود را در پیام دریافتی از KDCدیدند می توانند با اطمینان محاوره خود را آغاز کنند . داده های که در ادامه نشست بین طرفین مبادله می شوند با کلید نشست یعنی Ksرمز خواهند شد.

احراز هویت با پروتکل آتوی–ریس



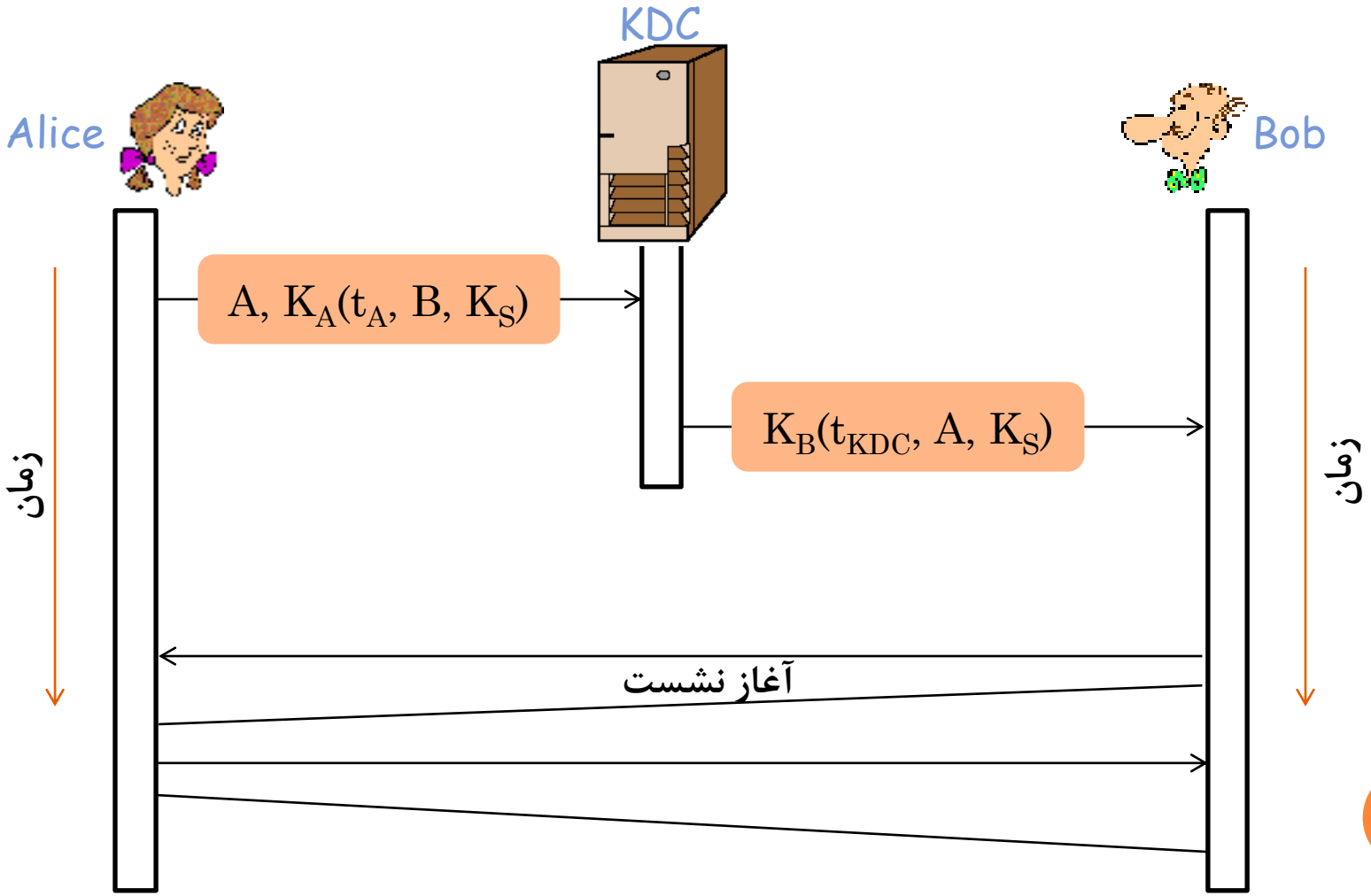
احراز هویت با «قورباغه دهن گشاد»

- تنها اشکال پروتکل احراز هویت ابتدایی، مشکل حمله تکرار بود
- با وجود مهر زمان این مشکل برطرف شد
- ساعت آلیس و باب باید با ساعت KDC تنظیم باشد
- مهر زمان دقتی در حد میلی ثانیه دارد و عمر پیام ها در حدود چند دقیقه
- برای مقاوم شدن این پروتکل در برابر حمله تکرار باب باید تا چند دقیقه مهر زمان را در حافظه خود نگهدارد

احراز هویت با «قورباغه دهن گشاد»

- آلیس پارامترهای ta مهر زمان Bشناسه کاربردی باب و Ksکلید نشست را با کلید متقارن خود رمز کرده و برای KDCمی فرستد با این اطمینان که هیچکسی قادر به رمز گشایی و سوء استفاده از آن نخواهد بود. (پارامتر زمان یا taتضمین می کند که پیام های تکراری و تاریخ گذشته به kdcتحویل نشود.)
- مرکز kdcپس از از رمز گشاایی قسمت دوم ، کلید نشست و شناسه کاربری باب و مهر زمان را استخراج و اگر اعتبار پیام هنوز باقی باشد ، Ksکلید نشست Aشناسه کاربری آلیس و مهر زمان جدیدی tkdcرا در یک ساختمان داده واحد قرار داده و کل آن را پس از رمز نگاری با کلید سری باب ، براب او می فرستد . باب پس از رمز گشایی پیام ، ابتدا اعتبار زمانی پیام را بررسی کرده و در صورت اعتبار ، نشست بین آلیس و باب با کلید نشست ksبرقرار خواهد شد.

احراز هویت با «قورباغه دهن گشاد»



احراز هویت با «کربروس»

مولفه های اصلی کربروس

1. Authentication Server : هر کاربر در اولین مرحله، باید فرآیند Login را طی کند تا هویت خود را به سرویس دهنده اثبات کند.
2. Ticket Granting Server : این سرور برای دریافت هر نوع سرویس از سرویس دهنده ها در سطح شبکه «بلیط» صادر می کند.
3. Server : این سرور نهایی پس از دریافت «بلیط» سرویس های خاصی را به مشتری ارائه می کند.

- سگ سه سر افسانه یونانی : محافظان دروازه های جهنم!
- سرها نماد:
  - Authentication
  - Accounting
  - Audit
- اکرچه در عمل تنها احراز هویت اعمال شد.

اجزاء Kerberos

- Authentication Server (AS)
- Ticket-Granting Server (TGS)
- Destination Server

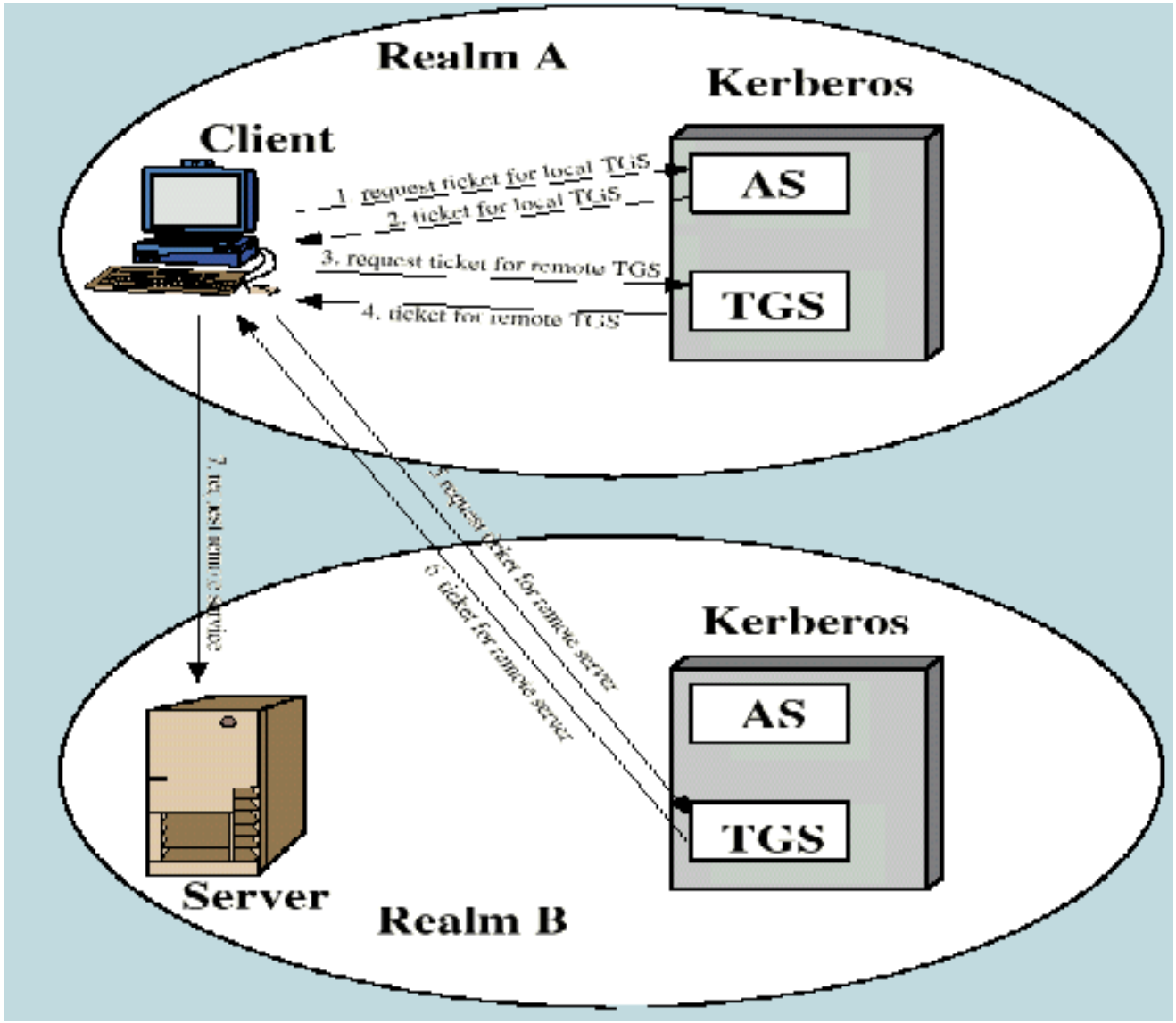
آشنایی با چند اصطلاح

احراز هویت روش یا مکانیزمی است که براساس آن هر مو جودیت (مثل یک پروسه یا شخص) بررسی می کند که شریک او در یک ارتباط (یعنی مو جودیت طرف مقابل) ، همانی است که ادعا می کند یا یک اخلال گر ثالث است که خود را به جای طرف واقعی جا زده است.

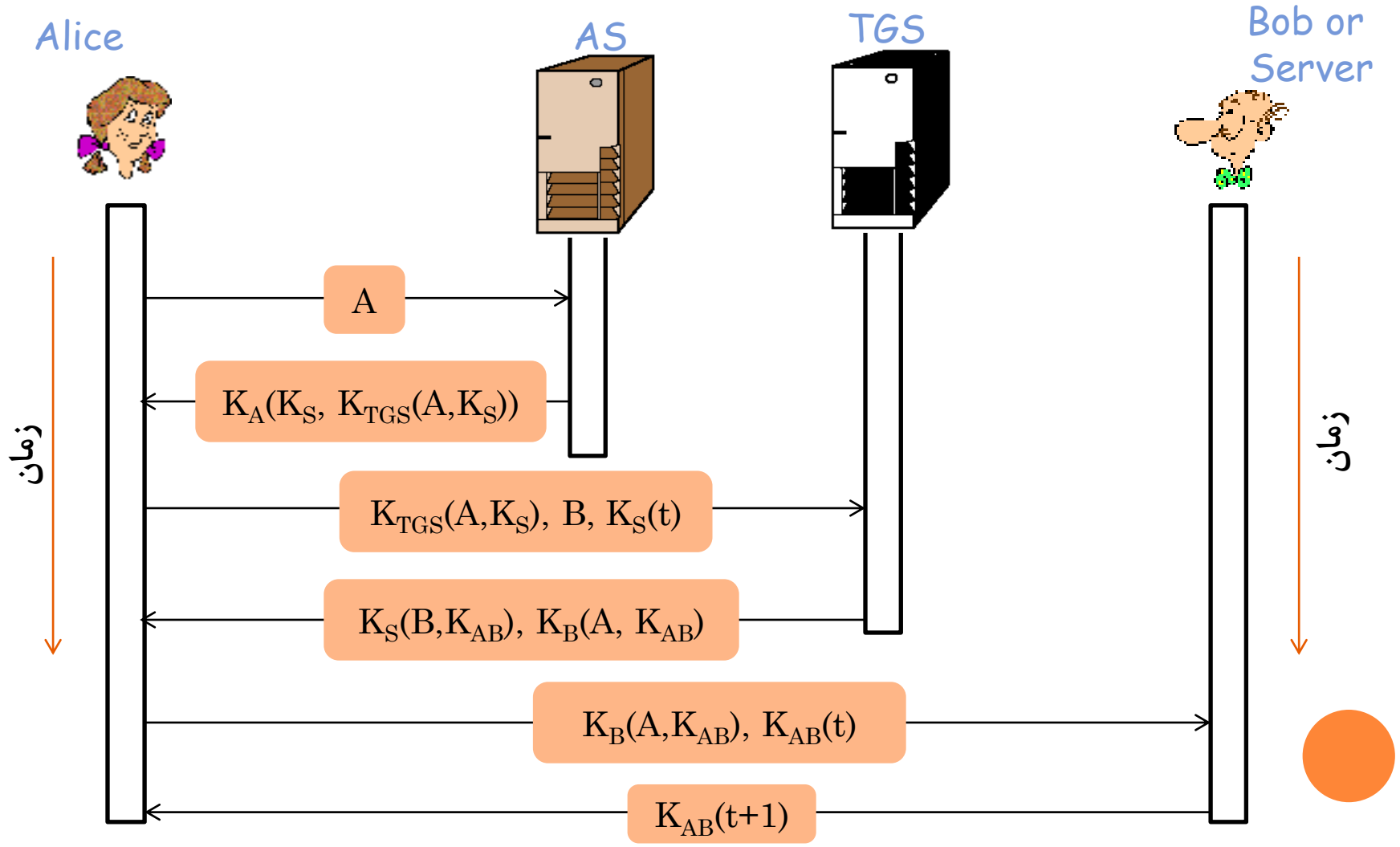
واژه مخفف AAA

- اولین A (تائید اعتبار) روشی که بر اساس آن ، هویت حقیقی یا حقوقی کاربران خود را اثبات می کنند .
- دومین A (تعیین سطح دسترسی) روشی که بر اساس آن مشخص می شود پروسه ای که هویت واقعی آن احراز شده ، مجوز انجام چه کارها و عملیاتی را دارد.
- سومین A (حسابداری) روشی که بر اساس مشخص می شود پروسه چه سهمی از منابع سیستمی و خدمات را می برد و آیا در ازای دریافت سهم خدمات ، حق و حساب آن را پرداخت کرده است ؟

Kerberos عملکرد



احراز هویت با «کربروس»



احراز هویت با «کربروس»

- روال دریافت سرویس در سیستم کربروس به شرح زیر است:
- آلیس همانند تمامی کاربران دیگر دارای کلید سری است که فقط سرویس دهنده ی AS از آن با خبر است لذا در اولین مرحله از ورود به سیستم آلیس باید از طریق ایستگاه کاری خود" Login " کند. این کار با ارسال آشکار شناسه ی کاربری آلیس (A) برای AS شروع می شود.
- سرویس دهنده ی AS دو قسمت Ks و Ktgs(A,Ks) را برای آلیس تولید کرده و پس از رمزنگاری با کلید سری او، برایش پس می فرستد:

  - Ks : یک کلید نشست
  - Ktgs(A,Ks) :بلیطی برای مراجعه به سرویس دهنده ی TGS
  - Ktgs :کلید سری سرویس دهنده ی TGS

- آلیس پیام دریافتی از ASرا با کلید سری خود رمزگشایی کرده و کلید نشست و بلیط خود را از درون آن استخراج می کند.سپس بلافاصله شا کلید خود(یعنی Ka) را از درون حافظه پاک می کند تا بیش از کسری از ثانیه در حافظه اصلی باقی نماند. حال هر زمان که آلیس بخواهد به هر کدام از سرویس دهنده های شبکه تقاضای سرویسی بدهد باید بلیط خود یعنی Ktgs(A,Ks) را به همراه شناسه ی سرویس دهنده مورد نظرش (B) برای سرویس دهنده ی TGS بفرستد. در ضمن او باید تاریخ و زمان صدور پیام را به کمک کلید نشست رمز کرده و نتیجه را به عنوان قسمت سوم (Ks(t)) ضمیمه کند.

احراز هویت با «کربروس»

- سرویس دهنده ی TGS ابتدا با کلید سری خود، بلیط را رمزگشایی کرده و شناسه ی کاربردی الیس و کلید نشست را از درون آن استخراج می نماید.
- سپس با کلید نشست قسمت سوم یعنی Ks(t) را رمزگشایی کرده و زمان صدور پیام را بررسی می کند، مبادا پیام، قدیمی و تکراری باشد. در صورتی که تازگی پیام محرز شد، مشروط برآنکه آلیس مجوز دریافت سرویس درخواستی را داشته باشد، برای آلیس و سرویس دهنده یک کلید نشست Kab تولید و آن را به همراه شناسه B با کلید Ks رمز کرده تا قسمت Ks(B,Kab) به دست آید. سپس قسمت دیگری به صورت Kb(A,Kab) تولید می کند که در حقیقت بلیط آلیس برای مراجعه به سرویس دهنده ی B است.این دو قسمت در پیام چهارم برای آلیس ارسال می شود تا آلیس بتواند با این بلیط از سرویس دهنده، در خواست سرویس کند.
- برای این کار آلیس ابتدا با کلید Ks قسمت Ks(B,Kab) را از رمز خارج کرده و پس از بررسی درستی نام سرویس دهنده، کلید مشترک Kab را از درون آن استخراج می کند.
- آلیس بلیط دریافتی خود یعنی Kb(A,Kab) را به عنوان قسمت اول و نتیجه ی رمزنگاری شده ی تاریخ و زمان صدور درخواست را به عنوان قسمت دوم برای سرویس دهنده می فرستد.
- سرویس دهنده از درون بلیط، هویت واقعی متقاضی و کلید نشست Kab را استخراج کرده و با Kab،مهر زمان را رمزگشایی می کند. در صورت تازه بودن درخواست،با افزایش یک واحدی مهر زمان (t) آن را برای آلیس پس می فرستد.با این کار هویت سرویس دهنده اثبات می شود چرا که فقط B می توانسته بلیط خود را رمزگشایی و از درون آن Kab را استخراج و مهر زمان را بررسی کرده و آن را پاسخ بدهد.

پس از پایان مرحله ششم، آلیس و سرویس دهنده قادر خواهند بود با کلید مشترک Kab یک نشست مطمئن و رمزنگاری شده برقرار کنند.

احراز هویت با «کربروس»

- «بلیط» ها به نام صادر می شوند(هویت متقاضی به صراحت در درون بلیط است)
- امکان حمله تکرار منتفی است زیرا همه پیام های ارسالی به سرورها، دارای مهر زمانی هستند
- اگر ترودی پیام سوم را گوش کرده و قبل از پایان مهر زمانی آن را تکرار کند، TGS هم به اشتباه پیام چهارم را به ترودی تحویل می دهد (اما ترودی K_S را ندارد ☺)
- هر سرویس هویتی مستقل داشته و «بلیط» جداگانه ای برای آن صادر می شود
- اما از «بلیط» AS به دفعات می توان با TGS ارتباط گرفت
- اگر کاربر سیستم خود را عوض کند همه چیز از نو شروع شود («بلیط» براساس ID کاربر و آدرس ماشین است)

کربروس

- احراز هویت بر اساس رمز نگاری کلید خصوصی
- 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
- مشهورترین کاربرد کربروس در Active Directory از شرکت میکروسافت است.

ویژگیهای عمومی کربروس

- عمومی بودن(Common)
  - در محیط توزیع شده همراه با سرورهای متمرکز و غیر متمرکز
- امنیت (Security)
  - ادعای اصلی
- اطمینان (Reliability)
  - اطمینان از فعال بودن همه سرویس ها برای کاربران مجاز.
- شفافیت (Transparency)
  - کاربران باید سیستم را همانند یک سیستم ساده "*شناسه و کلمه عبور*" ببینند.
- مقیاس پذیری (Scalability)
  - قابلیت کار با تعداد زیادی ماشین کاربر و کارگزار

ویژگیهای عمومی کربروس

چند تعریف

- دامنه: یک محدوده دسترسی را مشخص می کند. به نوعی معادل دامنه های تعریف شده در ویندوز یا Active Directory 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.

دیالوگ ساده احراز هویت-۰

درخواست خدمات توسط کارفرما از کارگزار:

1. ***Client → AS:*** $\mathbf{ID_{client} \,||\, Pass_{Client} \,||\, ID_{Server}}$
2. ***AS → Client:*** ***Ticket***
3. ***Client → Server:*** $\mathbf{ID_{client} \,||\, Ticket}$

$$\boldsymbol{Ticket = E_{Kserver}\,[ID_{client} \,||\, Addr_{client} \,||\, ID_{server}]}$$

AS : Authentication Server ***کارگزار احراز هویت***

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.

بررسی دیالوگ

- چرا آدرس کارفرما Client در بلیط ذکر میشود؟
  - در غیر این صورت هر شخصی که بلیط را از طریق شنود به دست آورد نیز میتواند از امکانات استفاده کند. اما اکنون تنها خدمات به آدرس ذکر شده در بلیط ارایه میشود.
    - مشکل جعل آدرس

- چرا شناسه مشتری ID_{client} در گام سوم به صورت رمز نشده ارسال میشود؟
  - زیرا این اطلاعات به صورت رمزنگاری شده در بلیط وجود دارد.
  - اگر شناسه با بلیط مطابقت نداشته باشد خدمات ارایه نمیشوند.

مشکلات دیالوگ ساده احراز هویت-۰

- ناامنی
  - ارسال کلمه عبور بدون رمزگذاری (بشکل متن واضح)
  - امکان حمله تکرار
- ناکارآیی
  - لزوم تقاضای بلیط جدید برای هر خدمات

استفاده مجدد از بلیط ها

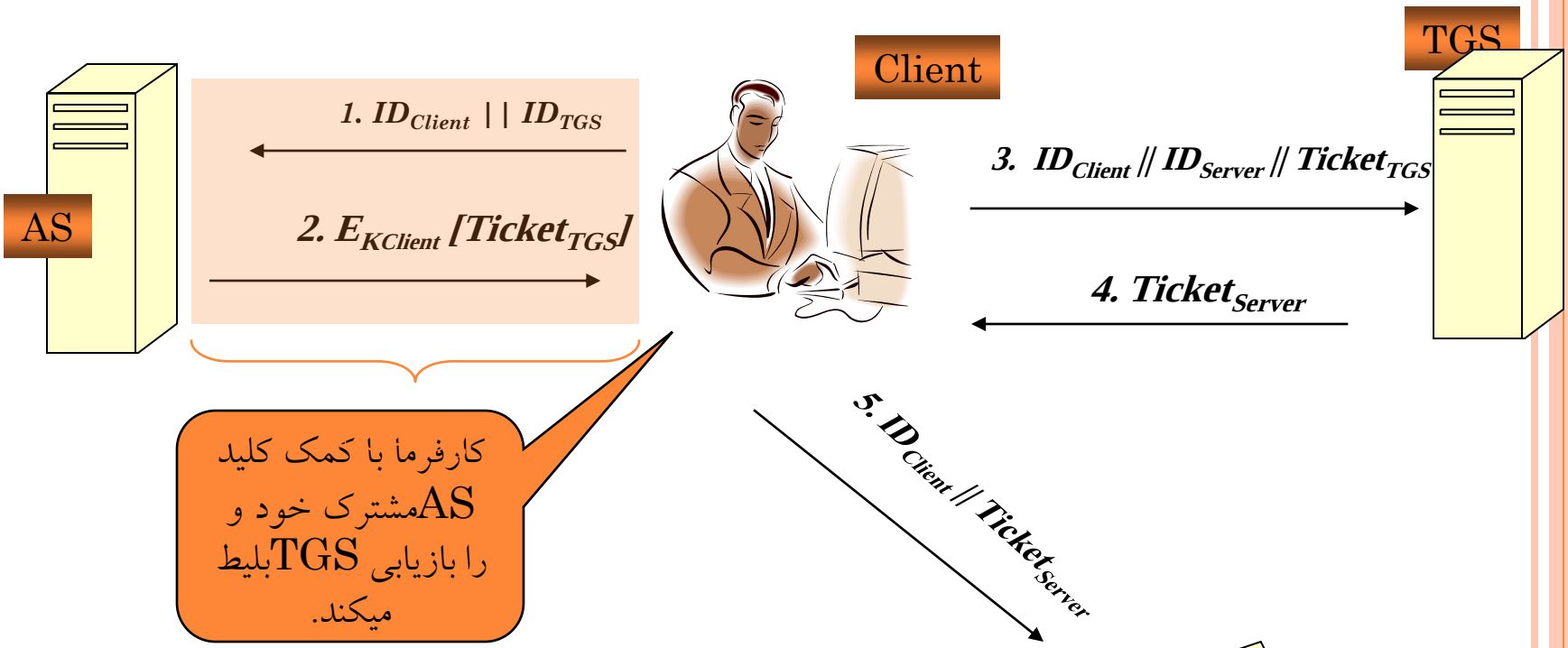
- چرا استفاده مجدد از بلیط ها (Tickets) اهمیت دارد؟
  - جلوگیری از تایپ مجدد کلمه عبور در یک بازه زمانی کوتاه
  - شفافیت احراز هویت
    - کاربر متوجه فرآیندهای هویت شناسی نمی شود.

افزایش ایمنی-دیالوگ ۱

- استفاده از یک کارگزار جدید با نام کارگزار اعطا کننده بلیط
  - TGS: Ticket Granting Server
- کارگزار احراز هویت، AS ، کماکان وجود دارد.
  - بلیط "اعطاء بلیط" ticket-granting ticket توسط آن صادر می شود.
- اگرچه بلیطهای اعطاء خدمات توسط TGS صادر میشوند.
  - بلیط "اعطاء خدمات" service-granting ticket
- اجتناب از انتقال کلمه عبور با رمز کردن پیام کارگزار احراز هویت (AS) به کارفرما

افزایش ایمنی -دیالوگ ۱

$$Ticket_{Server} = E_{K_{Server}} [ID_{Client} \| Addr_{Client} \| ID_{Server} \| Timestamp_2 \| Lifetime_2]$$



$$Ticket_{TGS} = E_{K_{TGS}} [ID_{Client} \| Addr_{Client} \| ID_{TGS} \| Timestamp_1 \| Lifetime_1]$$

افزایش ایمنی -دیالوگ ۱

- پیامهای شماره یک و دو به ازاء هر جلسه Log on رد و بدل میشوند.
- پیامهای شماره سه و چهار به ازاء هر نوع خدمات رد و بدل میشوند.
- پیام شماره پنج به ازاء هر جلسه خدمات رد و بدل میشود.

1. $Client \rightarrow AS: ID_{Client} \,||\, ID_{TGS}$
2. $AS \rightarrow Client: E_{KClient}[Ticket_{TGS}]$
3. $Client \rightarrow TGS: ID_{Client} \,||\, ID_{Server} \,||\, Ticket_{TGS}$
4. $TGS \rightarrow Client: Ticket_{Server}$
5. $Client \rightarrow Server: ID_{Client} \,||\, Ticket_{Server}$

محتوی بلیط ها

بلیط اعطای بلیط :

$$\text{Ticket}_{TGS} = E_{K_{TGS}} [ID_{Client} \| Addr_{Client} \| ID_{TGS} \| Timestamp_1 \| Lifetime_1]$$

بلیط اعطای خدمات :

$$\text{Ticket}_{Server} = E_{K_{Server}} [ID_{Client} \| Addr_{Client} \| ID_{Server} \| Timestamp_2 \| Lifetime_2]$$

ویژگی های دیالوگ ۱

- دو بلیط صادر شده ساختار مشابهی دارند. در اساس به دنبال هدف واحدی هستند.
- رمزنگاری $Ticket_{TGS}$ جهت احراز هویت
  - تنها کارفرما می تواند به بلیط رمزشده دسترسی پیدا کند.
- رمز نمودن محتوای بلیطها تمامیت (Integrity) را فراهم میکند.
- استفاده از مهر زمانی (Timestamp) در بلیطها آنها را برای یک بازه زمانی تعریف شده قابل استفاده مجدد میکند.
- هنوز از آدرس شبکه برای احراز هویت بهره میگیرد.
  - چندان جالب نیست زیرا آدرس شبکه جعل (Spoof) میشود.
  - با این حال، درجه ای از امنیت مهیا می شود

نقاط ضعف دیالوگ ۱

- مشکل زمان اعتبار بلیطها:
  - زمان کوتاه : نیاز به درخواست های زیاد گذرواژه
  - زمان بلند : خطر حمله تکرار
- هویت شناسی یکسویه : عدم احراز هویت کارگزار توسط کارفرما
  - رسیدن درخواست ها به یک کارگزارغیرمجاز

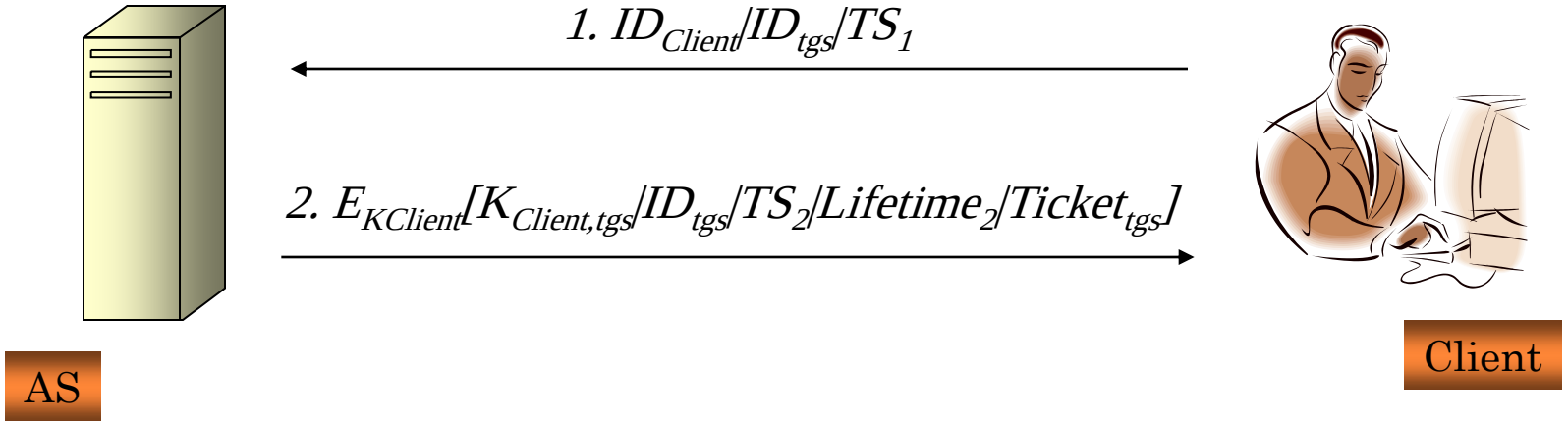
کربروس نسخه ٤

- توسعه یافته پروتکل های قبلی است
- مشکل حمله تکرار حل شده است.
- احراز هویت دو جانبه (mutual) برقرار میشود.
  - کارگزاران و کارفرمایان هردو از هویت طرف مقابل اطمینان حاصل میکنند

مقابله با حمله تکرار

- کارگزار یا TGS باید اطمینان حاصل نمایند که کاربر بلیط همان کسی است که بلیط برای او صادر شده.
- مفهوم جدیدی به نام اعتبار نامه (Authenticator) ابداع شده است:
  - علاوه بر بلیط ها
  - از مفهوم کلید جلسه بهره میجوید

کربروس نسخه ٤: بررسی الگوریتم-١

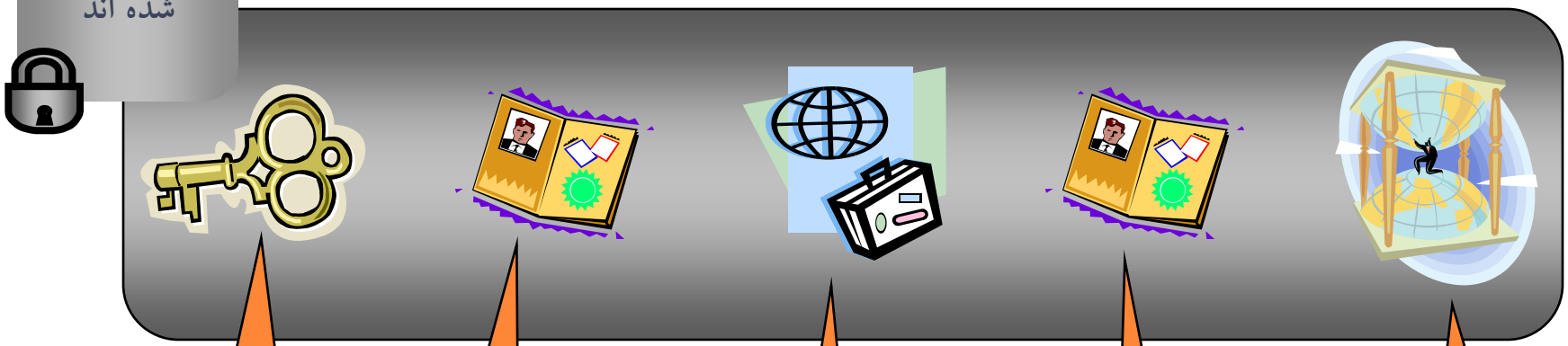


$$Ticket_{tgs}=E_{Ktgs}[K_{Client,tgs}/ID_{Client}/Addr_{Client}/ID_{tgs}/TS_2/Lifetime_2]$$

بلیط TGS

تمامی با کلید رمز TGS شده اند

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



کلید جلسه بین کارفرما وTGS

شناسه کارفرما

آدرس کارفرما

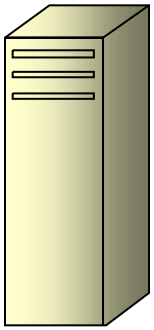
شناسه TGS

مهر زمانی و دوره اعتبار بلیط

نتایج این مرحله برای کارفرما

- بدست آوردن امن بلیط "اعطاء بلیط" از AS
- بدست آوردن زمان انقضای بلیط(TS_2)
- بدست آوردن کلید جلسه امن بین کارفرما و TGS

بدست آوردن بلیط "اعطاء خدمات"



Server

٣. $ID_{server} / Ticket_{tgs} / Authenticator_{Client}$

٤. $E_{K_{Client,tgs}} [K_{Client,server} / ID_{server} / TS_4 / Ticket_{server}]$

Client

$Ticket_{Server} =$

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

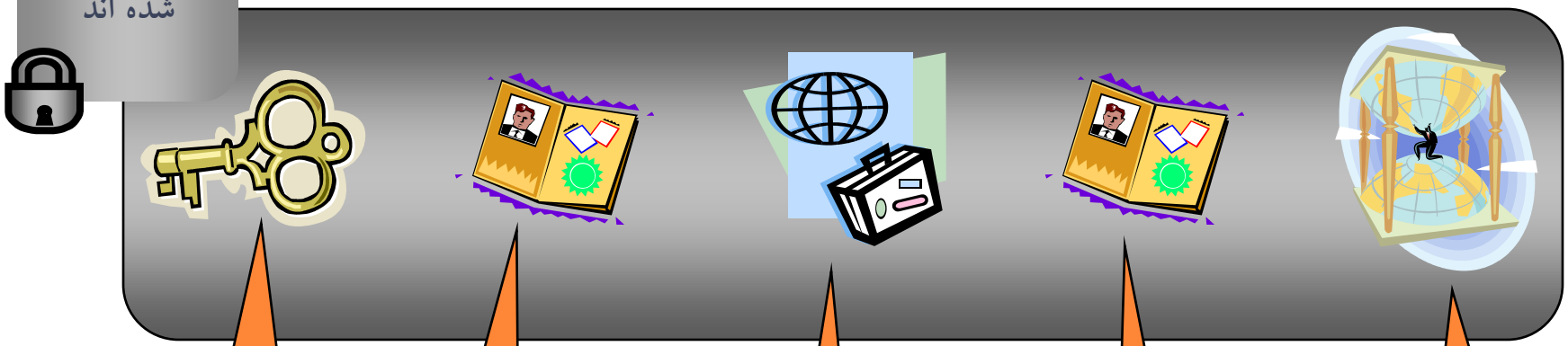
$Authenticator_{Client} =$

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

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

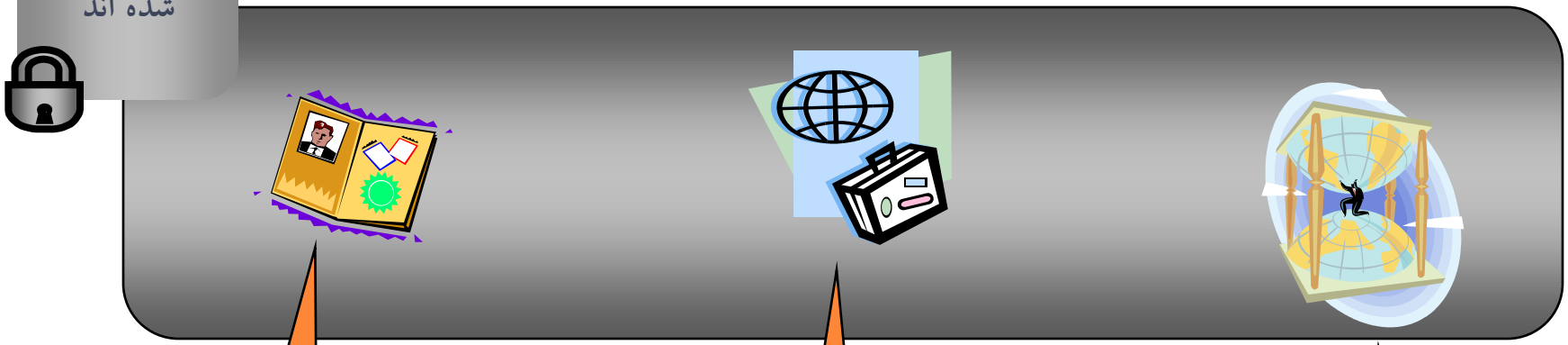
شناسه TGS

مهر زمانی و دوره اعتبار بلیط

اعتبار نامه کارفرما

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

تمامی با کلید جلسه رمز شده اند



شناسه کارفرما

آدرس کارفرما

مهر زمانی

نتایج این مرحله برای کارفرما

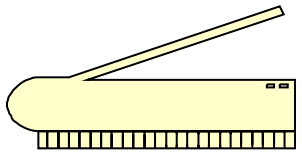
- جلوگیری از حمله تکرار با استفاده از یک اعتبار نامه (*Authenticator*) یکبار مصرف که عمر کوتاهی دارد.
- بدست آوردن کلید جلسه برای ارتباط با سرور V

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: $E_{K_c}\left[K_{c,tgs} \| ID_{tgs} \| TS_2 \| Lifetime_2 \| Ticket_{tgs}\right]$

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

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

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

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

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

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

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

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

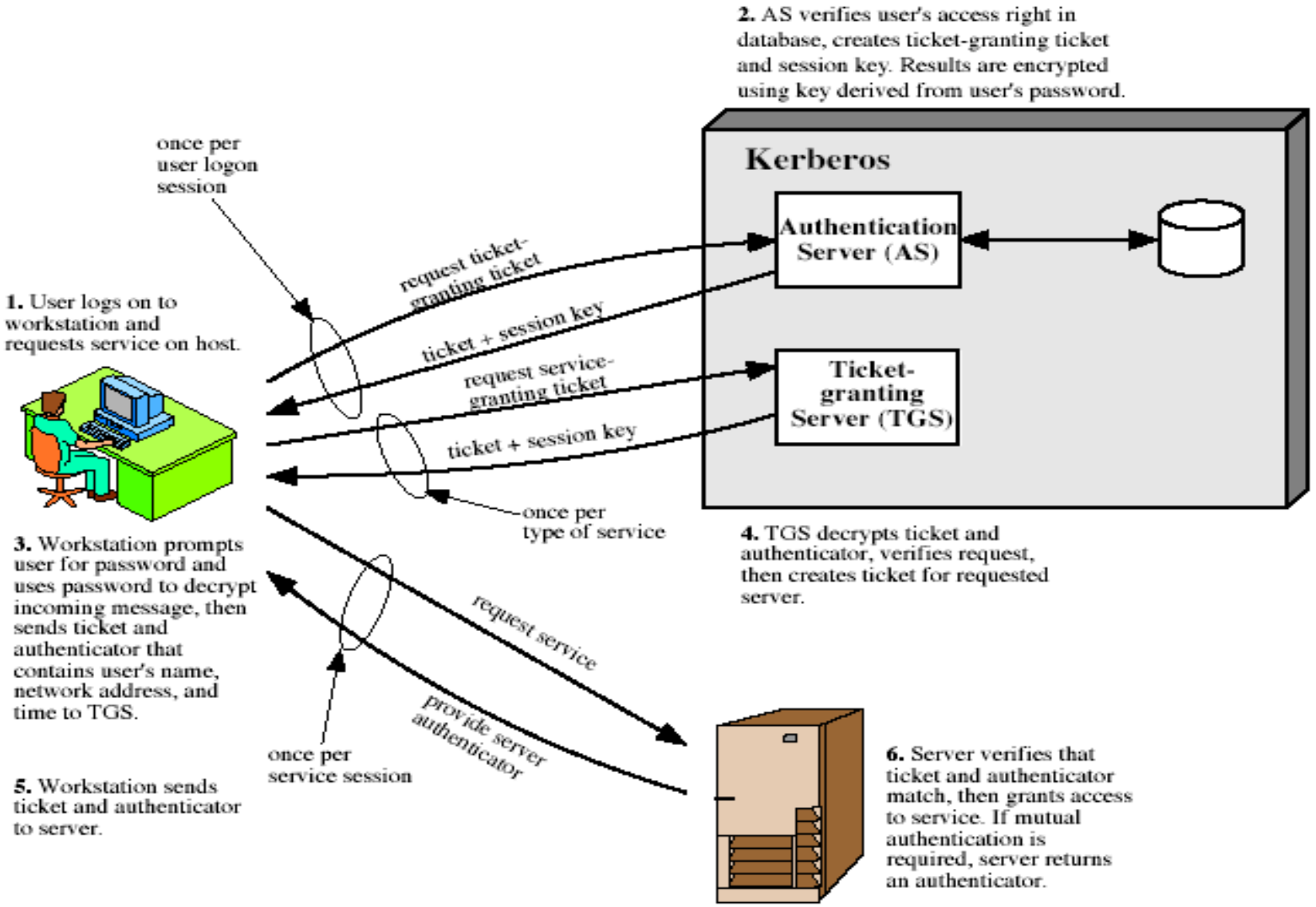
(5) C → V: $Ticket_v \| Authenticator_c$

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

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

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

کربروس نسخه ٤: شمای کلی



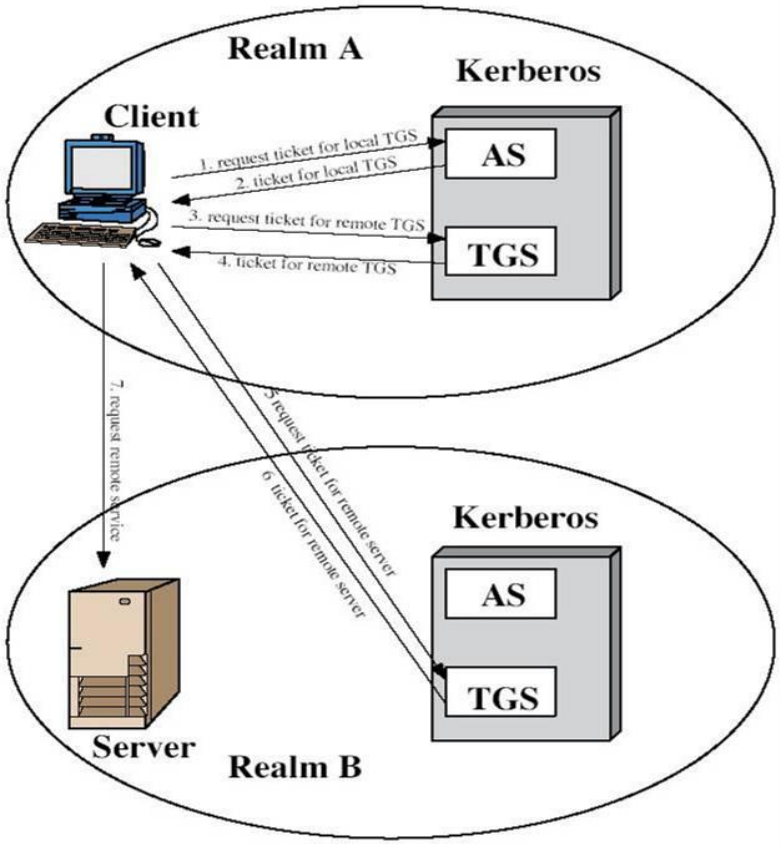
قلمرو کربروس

- 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.
- کارگزار کربروس با هر کارگزار کاربردی کلیدی مخفی به اشتراک گذاشته است.
- معمولاً هر قلمرو معادل یک حوزه مدیریتی میباشد.

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.
  - استاندارد جدید RFC 4120 باز بینی مجدد کربروس در سال ۲۰۰۵

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 آدرس شبکه(مثلا OSI یا IP) استفاده کرد
- محدود بودن زمان اعتبار بلیطها
  + در نسخه ۵ این محدودیت وجود ندارد

مشکلات KERBEROS V4 و نحوه رفع آنها در نسخه ۵

- امکان انتقال اعتبار یک کاربر به یک سرور دیگر وجود ندارد

+ در نسخه ۵ اجازه داده می شود که مثلا کاربر به سرور چاپ یک فایل را معرفی کند تا فایل با اعتبار آن کاربر توسط سرور چاپ، چاپ شود (در یک ماشین دیگر)

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

+ در نسخه ۵ این مشکل حل شده است.

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

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

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

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

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

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

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

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

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

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

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

- **نکات زیر در مورد کربروس شایان ذکر است:**
- **بلیطها به نام صادر می شوند بدین معنا که هویت متقاضی بلیط به صراحت درون بلیط درج می شود. بدین ترتیب امکان آن که کسی با عالی ترین مجوز بتواند برای افراد بدون مجوز بلیط تهیه کند، منتفی است. در ضمن اگر شخص ثالثی مثل ترودی بلیطی را استراق سمع کرد گذشته از آن که قادر به استخراج کلید نشست از درون آن نیست نمی تواند آن را به نام خودش برای سرویس دهنده بفرستد.**
- **امکان حمله ی تکرار منتفی است زیرا تقاضاهای ارسالی به سرویس دهنده ی صدور بلیط و سرویس دهنده ی نهایی، مهر زمان دارد.**
- **فرض کنید شخص ثالثی مثل ترودی پیام سوم را استرق سمع کند و قبل از آنکه اعتبار زمانی آن منقضی شود آن را تکرار کند. سرویس دهنده ی TGS با توجه به آنکه هنوز زمان اعتبار پیام منقضی نشده، بدون اعتنا به تکراری بودن آن، پیام چهارم را به ترودی تحویل خواهد داد ولی هیچ تهدیدی وجود ندارد زیرا ترودی کلید Ks را برای رمزگشایی Ks(B,Kab) ندارد و طبعا قادر نخواهد بود قسمت Kab(t) را برای پیام پنجم تولید کند.**
- **در کربروس گذشته از آنکه هیچ کلمه ی عبوری به صورت آشکار بر روی خط منتقل نمی شود شاه کلید بیش از چند میلی ثانیه در حافظه ی ماشین نخواهد ماند.**

- نکته ی ارشمند در این ساختار آن است که AS فقط کاربران را می شناسد و کلید آنها را در اختیار دارد و TGS فقط با سرویس دهنده ها سر و کار دارد و برای آنها بلیط صادر می کند.

- سرویس دهنده ی AS و TGS از دیدگاه منطقی جدا هستند ولی در عمل می توانند بر روی یک ماشین واحد اجرا شوند.

- سرویسهای مختلفی که بر روی یک ماشین واحد ارائه می شوند هر کدام هویت مستقلی دارند و بلیط جداگانه ای بر هر کدام صادر خواهد شد.

- هر پروسه بر روی ماشین آلیس که نیازمند سرویسی باشد باید مستقلا بلیط تهیه کند. به عبارت دقیقتر TGS بلیطی را برای یک پروسه جهت دریافت صرفا یک سرویس خواص صادر می کند.

- با بلیطی که توسط سرویس دهنده ی ASصادر می شود می توان به دفعات از TGS تقاضای صدور بلیط از سرویس دهنده های مختلف داشت.

- پس از پایان مرحله ی ششم، بین پروسه ی مشتری و پروسه ی سرویس دهنده،نشستی شکل می گیرد که تا پایان رسمی نشست، نیاز به احراز هویت ندارد و از آنجا که با کلید نشست رمزنگاری می شود، نشستی امن است.

- در کربروس بلیطها برحسب نام شناسه کاربری و آدرس ماشین در شبکه صادر می شوند یعنی آلیس حتی از یک ایستگاه کاری به ایستگاه کاری دیگر تغییر موقعییت بدهد قادر به استفاده از بلیط های قبلی خود نیست و باید از نوع تمام مراحل احراز هویت را تکرار کند. بدین ترتیب تغییر مبدا و مقصد یک نشست فعال، از یک ماشین به ماشین دیگر ممکن نیست.

- در کربروس، سرویس دهنده TGS تقاضای صدور بلیط را از لحاظ جواز دسترسی ارزشیابی می کند لذا یک بانک اطلاعاتی خاص، سطوح دسترسی کاربران را مشخص کرده است؛ همین کار در سرویس دهنده های نهایی نیز انجام می شود.

- در نسخه ی پنجم کربروس، امکان استفاده از هر روش رمزنگاری کلید متقارن وجود دارد، در حالی که نسخه های قبلی عملا به DES وابسته بودند.

- در کربروس هر کاربری در خلال مراحل اول و دوم،ابتدا احراز هویت و سپس در شبکه وارد می شود، بلیطی به شکل Ktgs(A , Ks) دریفت می دارد که برای ساعتها معتیر است و او می تواند به دفعات از این بلیط بهره بگیرد و با شروع از مرحله ی سوم برای دریافت خدمات از هر سرویس دهنده ی دیگر شبکه، تقاضا بدهد. برای این کار کافی است در پیام سوم، نام سرویس دهنده ی مورد نظر را درج شود؛ در پاسخ،بلیطی بر خواهد گشت که مربوط به همان سرویس دهنده است.
- صدور بلیط یک سرویس دهنده برای کاربران بدین معنا نیست که کاربر می تواند ارائه بلیط به سرویس دهنده، هر سرویس دلخواهی را دریافت کند بلکه خود سرویس دهنده هم ممکن است کاربران را وادار به یک "Login" مجدد کند و دسترسی او را با ضوابط خاصی فراهم گردد.

- در نسخه ی ۵ از کربروس هر بلیط صادره توسط TGS دارای محتویات زیر است:
- Login Name (User ID) شناسه ی کاربردی/ شناسه ورود
- Server Name نام نمادین سرویس دهنده
- Client Host Network Address آدرس ماشین مشتری
- Session Key کلید نشست
- Ticket Lifetime زمان اعتبار بلیط
- Creation Timestamp مهر زمان

- **گوشه ای از برتریهای نسخه ی پنجم کربروس:**

- **برای آنکه سیستم کربروس قابلیت گسترش در سطح شبکه های بسیار بزرگ را داشته باشدسعی شده که کل شبکه به صورت سلسله مراتبی در قالب چندین قلمرو تقسیم بندی شود. هر قلمرو برای خودش یک سرویس دهنده ی AS و یک TGS دارد که بار کاربران قلمروی خودش را به دوش می کشد،ولی این امکان وجود دارد که مسئول شبکه، قلمروها و سرویس دهنده های TGS را به گونه ای تنظیم و پیکربندی کند که یک سرویس دهنده ی TGS از یک قلمرو اجازه داشته باشد برای همه یا تعدادی از سرویس دهنده های قلمروی دیگران بلیط صادر کند.**
- **امکان ایجاد اعتماد متقابل بین TGS در دو قلمرو، مستلزم آنست که بین آنها کلید مشترک تعریف شود. در نسخه ی پنجم، از وجود قلمروهای مختلف و تفویض صدور کلید به سرویس دهنده ی این نواحی به خوبی حمایت می شود.**

- **نسخه ی پنجم کربروس در محیطی با ۲۸۰۰۰۰ کاربر آزمایش شده و کارآیی آن به اثبات رسیده است.**

پیاده سازی های موجود کربروس

- دانشگاه MIT : اولین پیاده سازی کربروس که هنوز به عنوان مرجع مورد استفاده قرار می گیرد
- Heimdal : تنها پیاده سازی انجام شده در خارج آمریکا (سوئد)
- Active Directory : پیاده سازی ارائه شده توسط مایکروسافت که در RFC 1510 آمده است

با توجه به اینکه کربروس علاوه بر احراز هویت بررسی حقوق دسترسی را نیز انجام میدهد براحتی میتواند بعنوان یک فایروال کامل و دقیق عمل کند. در اینصورت این فایروال نه تنها از دسترسیهای غیر مجاز از خارج شبکه جلوگیری میکند بلکه بعنوان یک فایروال حتی دسترسیهای غیر مجاز داخل شبکه را نیز مسدود مینماید.