یا ذاالامن والامان

امنیت داده و شبکه

پروتکل کربروس

مرتضی امینی - نیمسال دوم ۱۴۰۰-۱۳۹۹

مرکز امنیت شبکه شریف
http://nsc.sharif.edu

فهرست

کربروس (Cerberus یا Kerberos)

- **Kerberos (Κέρβερος)**: بر گرفته از اسطوره یونانی
  - نام سگی سه‌سر که محافظ دروازه‌های عالم مردگان بود؛ نمی‌گذاشت زندگان مزاحم ارواح شده و ارواح از عالم مردگان خارج شوند.

- سرها نماد:
  - احراز اصالت (Authentication)
  - مجازشماری (Authorization)
  - حسابرسی (Accounting)

تاریخچه کربروس – ۱

- مبتنی بر پروتکل احراز اصالت نیدهام-شرودر (۱۹۷۸) و اصلاح شده آن توسط دنینگ و ساکو (۱۹۸۱).
  - کلید متقارن؛ استفاده از KDC؛ به کارگیری برچسب زمانی.
- نسخه ۱ الی ۳ کربروس در MIT به صورت داخلی.
- نسخه ۴ در سال ۱۹۹۰ به طور رسمی منتشر شد.
  - استفاده از DES برای رمزنگاری
  - دارای محدودیتها و اشکالات امنیتی فراوان

تاریخچه کربروس – ۲

- نسخه ۵ در ۱۹۹۳ به طور رسمی منتشر شد (RFC 1510).
- تا سال ۲۰۰۰، رمزنگاری در آمریکا «سلاح» محسوب می‌شد.
  - انتشار کد کربروس به خارج از آمریکا جرم بود.
  - دانشگاه KTH سوئد با الهام از مستندات نسخه ۴، نسخه‌ای از کربروس را با عنوان eBones منتشر ساخت.
- استاندارد ۵ کربروس (RFC 1510) در سال ۲۰۰۵ تحت عنوان RFC 4120 اصلاح شد.

تاریخچه کربروس – ۳

- سایر اصلاحات کربروس در ۲۰۰۵:
  - به کارگیری روشهای متنوع رمزنگاری و صحت (RFC 3961)
  - به کارگیری AES (RFC 3962)
- در سال ۲۰۰۷، MIT کنسرسیوم کربروس را تشکیل داد.
  - شامل شرکتهایی چون اراکل، اپل، گوگل، و مایکروسافت
  - نهادهای دانشگاهی نظیر KTH، استنفورد و MIT
- مایکروسافت با توجه به استفاده از کربروس در Active Directory، چندین بهبود عمده در آن داده است.

تاریخچه کربروس – ۴

- بهبودهای مایکروسافت:
  - به روز رسانی GSS-API در RFC 4121
  - مذاکره الگوریتمهای رمز در RFC 4537
  - امکان استفاده از زیرساخت کلید عمومی در کربروس (PKIINIT) در RFC 4556
  - OCSP برای PKINIT در RFC 4557
  - الگوریتمهای RC4-HMAC برای کربروس در RFC 4757
  - رمزنگاری خم بیضوی برای PKINIT در RFC 5349

تاریخچه کربروس – ۵

- سایر بهبودهای مایکروسافت:
  - قیود جدید برای نامها در RFC 6111
  - پشتیبانی از گمنامی در RFC 6112
  - پیش احراز اصالت در RFC 6113
  - استاندارد سازی نامگذاری در RFC 6806
- امروزه سه پیاده‌سازی عمده از کربروس وجود دارد: MIT، هایمدال (Heimdal) و مایکروسافت (Active Directory)

کربروس

- احراز اصالت بر اساس رمزنگاری متقارن برای محیط‌های توزیع شده
- به جای احراز اصالت در هر کارگزار به صورت توزیع شده، یک کارگزار خاص را به احراز اصالت اختصاص می‌دهیم.
- نسخه‌های ۴ و ۵ آن در حال استفاده هستند.

نیازمندیها/ویژگیهای عمومی کربروس

- امنیت (Security)
  - با شنود در شبکه، امکان جعل کاربر توسط مهاجم وجود نداشته باشد.
- اطمینان (Reliability)
  - اطمینان از دسترس‌پذیری کارگزار احراز اصالت کربروس با پشتیبانی از خدمت‌رسانی توزیع شده و کارگزار پشتیبان.
- پنهانی (Transparency)
  - کاربران باید سیستم را همانند یک سیستم ساده مبتنی بر گذرواژه ببینند.
- مقیاس‌پذیری (Scalability)
  - قابلیت کار با تعداد زیادی ایستگاه کاری و کارگزار با پشتیبانی از ساختاری پیمانه‌ای و توزیع‌شده.

ویژگیهای عمومی کربروس

- چند تعریف
  - **دامنه (realm):** یک محدوده دسترسی (محدوده مدیریتی) را مشخص می‌کند. به نوعی معادل دامنه‌های تعریف شده در ویندوز است.
  - **مرکز توزیع کلید:** معادل کارگزار کربروس است.
  - **عامل (Principal):** به سرویس‌ها، دستگاه‌ها، کاربران و کلیه عناصری که نیاز به شناساندن و احراز خود به کارگزار کربروس دارند، عامل گفته می‌شود.

کربروس

- برای معرفی کربروس به صورت گام به گام از پروتکلهای ساده شروع می‌کنیم و سعی می‌کنیم اشکالات هر یک را برطرف کنیم تا به کربروس برسیم.

فهرست

- مقدمه و نیازمندی‌ها
- **دیالوگ‌های ساده احراز اصالت**
- کربروس نسخه ۴
- کربروس نسخه ۵

دیالوگ ساده احراز اصالت

- فرض: بین کارگزار احراز اصالت (AS) و هر کارگزار سرویس (V) یک کلید مشترک وجود دارد.
- درخواست خدمات توسط کاربر از کارگزار سرویس:
  - **C→AS:** $ID_C \| P_C \| ID_V$
  - **AS→C:** Ticket
  - **C→V:** $ID_C \| Ticket$
  - $Ticket = E(K_v, [ID_C \| AD_C \| ID_V])$

C = client AS= authentication server

V =server P_C= password of user on C

AD_C = network address of C

K_v = secret encryption key shared by AS and V

بلیط

- در واقع نوعی گواهی است که هنگام ورود کاربر به دامنه کربروس به او داده می‌شود که بیانگر اعتبار او برای دسترسی به منابع شبکه است.

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 کارفرما ID_C در گام سوم به صورت رمز نشده ارسال می‌شود؟
  - زیرا این اطلاعات به صورت رمز شده در بلیط وجود دارد.
  - اگر شناسه با بلیط مطابقت نداشته باشد خدمات ارایه نمی‌شوند.

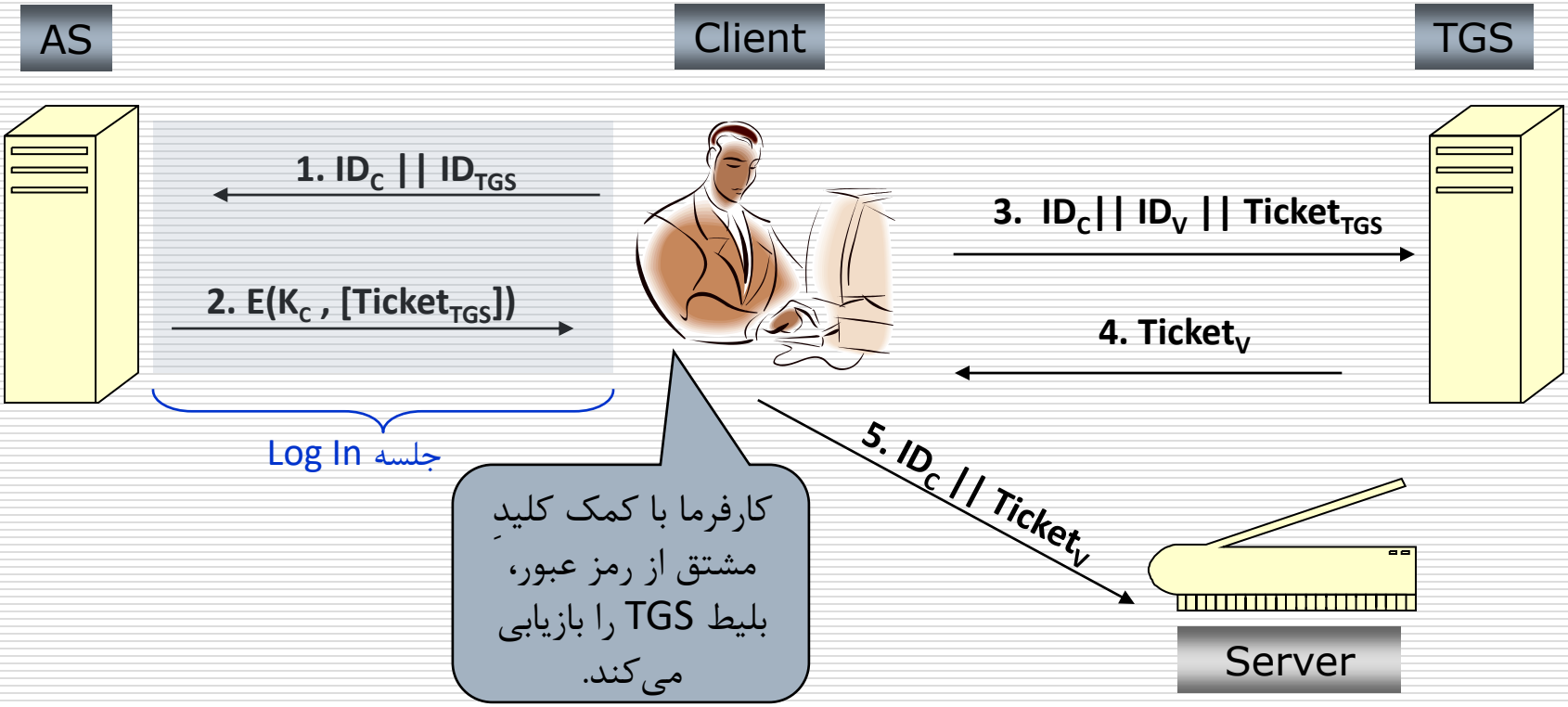
مشکلات دیالوگ ساده احراز اصالت

- ناامنی
  - ارسال کلمه عبور بدون رمزگذاری
  - امکان حمله تکرار (با شنود، جعل شناسه و جعل آدرس کاربر)
- ناکارآیی
  - لزوم احراز اصالت کاربر برای دریافت بلیط جدید برای هر خدمت

ارتقای امنیت–دیالوگ ۱

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

ارتقای امنیت–دیالوگ۱



$$Ticket_{TGS} = E\ K_{TGS}, [ID_C \| Addr_C \| ID_{TGS} \| Timestamp_1 \| Lifetime_1]$$

$$Ticket_V = E\ K_V, [ID_C \| Addr_C \| ID_V \| Timestamp_2 \| Lifetime_2]$$

ارتقای امنیت–دیالوگ۱

- پیام‌های شماره ۱ و ۲ به ازاء هر جلسه Log on رد و بدل می‌شوند.
- پیام‌های شماره ۳ و ۴ به ازاء هر نوع خدمات رد و بدل می‌شوند.
- پیام شماره ۵ به ازاء هر جلسه خدمات رد و بدل می‌شود.

1. **C→AS:** $ID_C \parallel ID_{TGS}$
2. **AS→C:** $E(K_c, Ticket_{TGS})$
3. **C→TGS:** $ID_C \parallel ID_V \parallel Ticket_{TGS}$
4. **TGS→C:** $Ticket_v$
5. **C→V:** $ID_C \parallel Ticket_v$

محتوی بلیط ها

- بلیط اعطای بلیط:

$$Ticket_{TGS} = E\ K_{TGS}, [ID_C \| Addr_C \| ID_{TGS} \| Timestamp_1 \| Lifetime_1]$$

- بلیط اعطای خدمات:

$$Ticket_V = E\ K_V, [ID_C \| Addr_C \| ID_V \| Timestamp_2 \| Lifetime_2]$$

ویژگی های دیالوگ ۱

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

نقاط ضعف دیالوگ ۱

- مشکل زمان اعتبار بلیط‌ها:
  - زمان کوتاه: نیاز به درخواست‌های زیاد گذرواژه
  - زمان بلند: خطر حمله تکرار

- احراز اصالت یکطرفه: عدم احراز اصالت کارگزار سرویس توسط کارفرما
  - رسیدن درخواست‌ها به یک کارگزار غیرمجاز به جای کارگزار اصلی ۷.

فهرست

- مقدمه و نیازمندی‌ها
- دیالوگ‌های ساده احراز اصالت
- **کربروس نسخه ۴**
- کربروس نسخه ۵

کربروس نسخه ۴

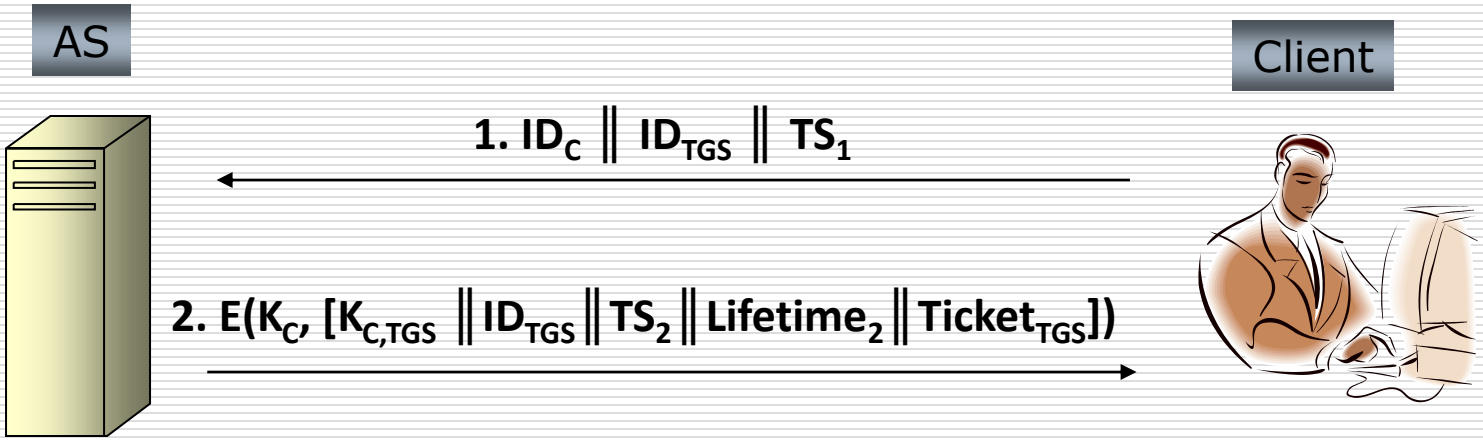
- توسعه یافته پروتکل‌های قبلی است.
- مشکل حمله تکرار حل شده است.
- احراز اصالت دو جانبه صورت می‌گیرد.
  - کارگزاران و کارفرمایان هر دو از اصالت هویت طرف مقابل اطمینان حاصل می‌کنند.

مقابله با حمله تکرار

- یک نیاز جدید: کارگزار یا TGS باید اطمینان حاصل نمایند که کاربرِ بلیط، همان کسی است که بلیط برای او صادر شده.

- مفهوم جدیدی به نام اعتبارنامه (Authenticator) ابداع شده است:
  - علاوه بر بلیط‌ها از مفهوم کلید جلسه بهره می‌جوید.

بدست آوردن بلیط اعطاء بلیط



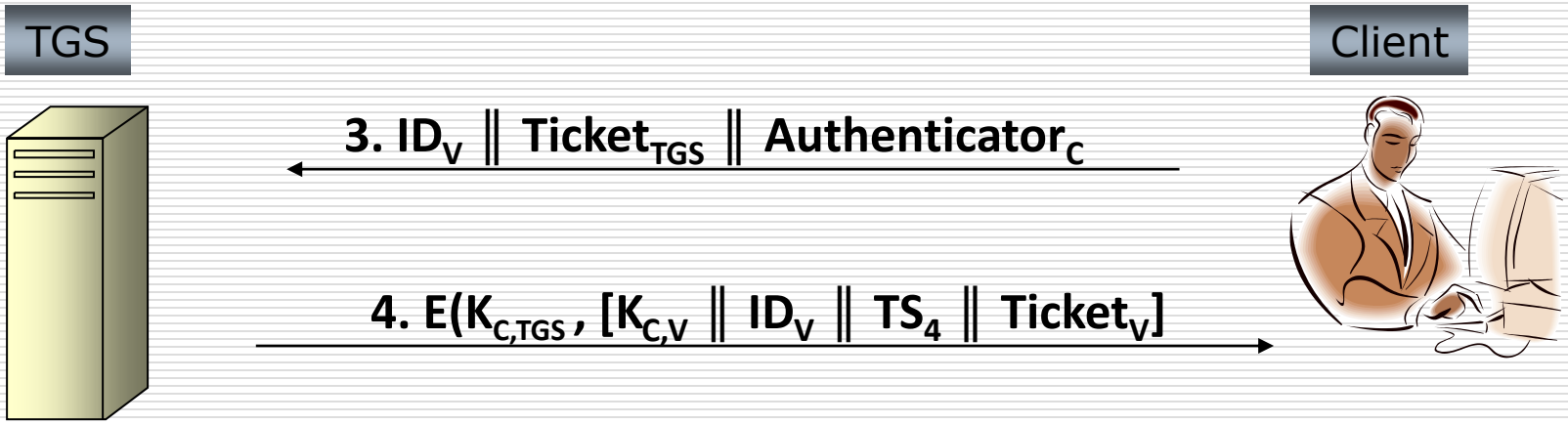
K_C: کلید مشتق شده از گذرواژه کاربر

$$Ticket_{TGS}=E(K_{TGS}, [K_{C,TGS} \| ID_C \| Addr_C \| ID_{TGS} \| TS_2 \| Lifetime_2])$$

بدست آوردن بلیط اعطاء 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



$$Ticket_V = E(K_V, [K_{C,V} \| ID_C \| Addr_C \| ID_V \| TS_4 \| Lifetime_4]$$

$$Ticket_{TGS} = E(K_{TGS}, [K_{C,TGS} \| ID_C \| Addr_C \| ID_{TGS} \| TS_2 \| Lifetime_2])$$

$$Authenticator_C = E(K_{C,TGS}, [ID_C \| Addr_C \| TS_3])$$

بدست آوردن بلیط اعطاء خدمات

- 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 کارگزار V

دستیابی به خدمات کارگزار

$Ticket_V = E(K_V, [K_{C,V} \| ID_C \| Addr_C \| ID_V \| TS_4 \| Lifetime_4])$

$Authenticator_C = E(K_{C,V}, [ID_C \| Addr_C \| TS_5])$

دستیابی به خدمات کارگزار

- نتایج این مرحله برای کارفرما
  - احراز اصالت کارگزار در گام ششم با برگرداندن پیغام رمزشده
  - جلوگیری از بروز حمله تکرار

کربروس نسخه ۴: 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

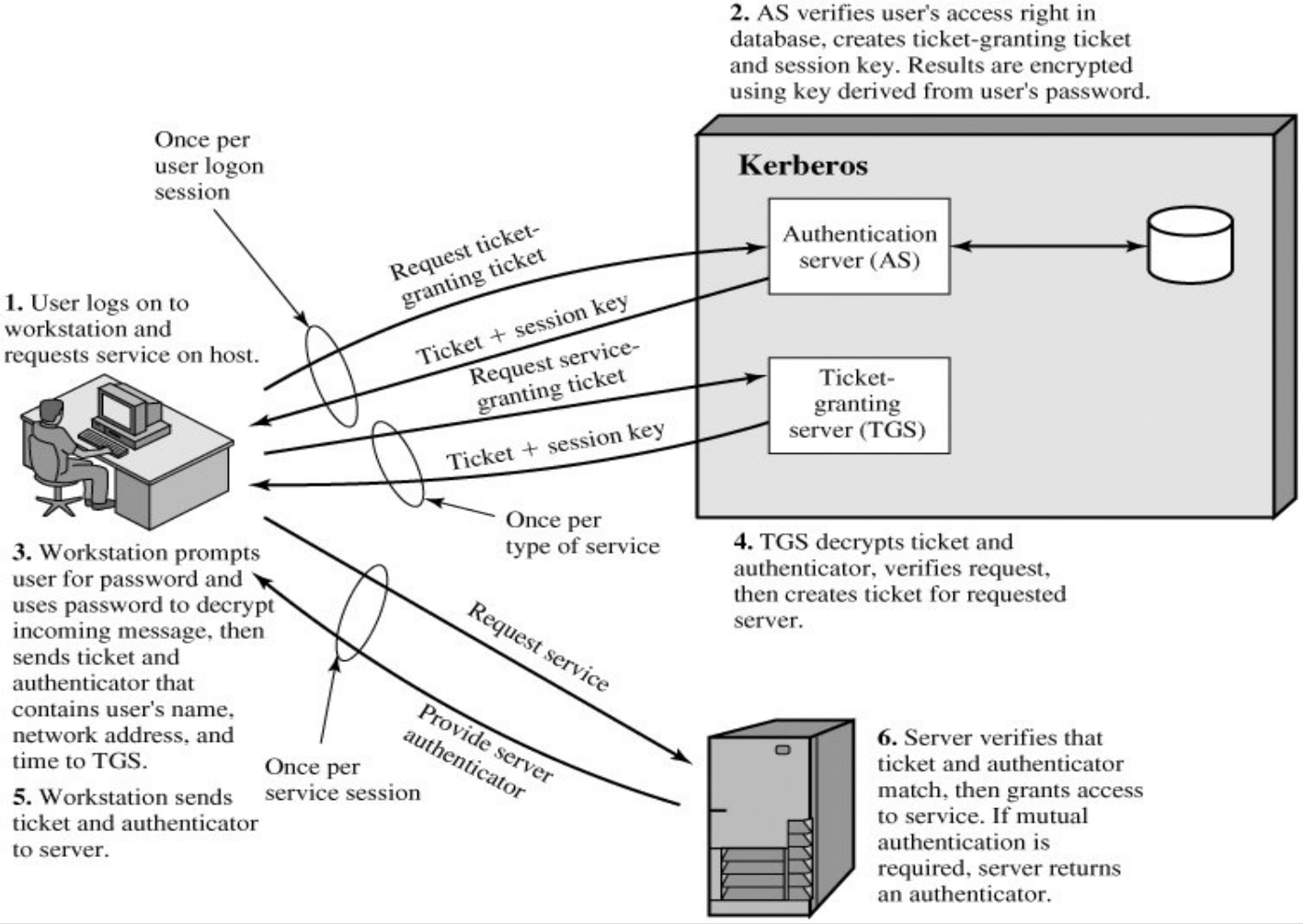
(5) C → V: $Ticket_v \| Authenticator_c$

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

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

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

کربروس نسخه ۴: شمای کلی



دامنه کربروس (realm)

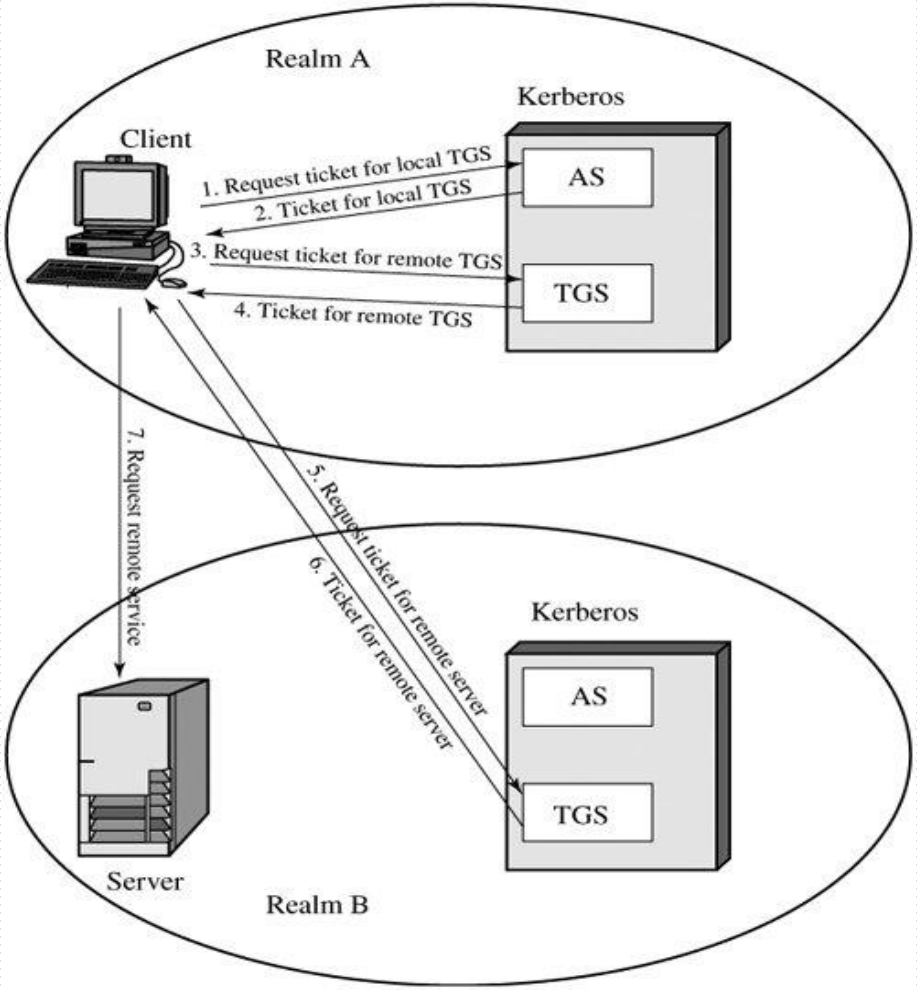
- دامنه کربروس از بخش‌های زیر تشکیل شده است:
  - کارگزار کربروس
  - کارفرمایان (کاربران)
  - سرویسها یا کارگزاران برنامه‌های کاربردی (Application Servers)
- کارگزار کربروس گذرواژه تمام کاربران را در پایگاه‌داده‌های خود دارد.
- کارگزار کربروس با هر کارگزار برنامه کاربردی کلیدی مخفی به اشتراک گذاشته است.
- معمولاً هر دامنه معادل یک حوزه مدیریتی است.

تعامل بین دامنه‌ای

- وجود بیش از یک دامنه کربروس
- نیاز به دریافت سرویس از کارگزاری در دامنه دیگر
- نیاز به وجود کلید مشترک بین هر دو کارگزار کربروس
- رویه پیشنهادی:
  - احراز اصالت کاربر توسط کارگزار کربروس
  - دریافت بلیط از TGS محلی برای ارتباط با TGS دامنه بیرونی
  - ارتباط با TGS دامنه بیرونی برای دریافت بلیط دریافت سرویس
  - ارائه بلیط به کارگزار سرویس دامنه بیرونی برای دریافت سرویس

احراز اصالت بین دامنه‌ای

- **C→AS:** $ID_C \| ID_{TGS} \| TS_1$
- **AS→C:** $E(K_C, [K_{C,TGS} \| ID_{TGS} \| TS_2 \| Lifetime_2 \| Ticket_{TGS}])$
- **C→TGS:** $ID_{TGSrem} \| Ticket_{TGS} \| Authenticator_C$
- **TGS→C:** $E(K_{C,TGS}, [K_{C,TGSrem} \| ID_{TGSrem} \| TS_4 \| Ticket_{TGSrem}])$
- **C→TGS_{rem}:** $ID_{Vrem} \| Ticket_{TGSrem} \| Authenticator_C$
- **TGS_{rem} → C :** $E(K_{C,TGSrem}, [K_{C,Vrem} \| ID_{Vrem} \| TS_6 \| Ticket_{Vrem}])$
- **C→V_{rem}:** $Ticket_{Vrem} \| Authenticator_C$



فهرست

- مقدمه و نیازمندی‌ها
- دیالوگ‌های ساده احراز اصالت
- کربروس نسخه ۴
- **کربروس نسخه ۵**

کربروس نسخه ۵

مشخصات

- در اواسط دهه ۱۹۹۰ مطرح شد.
- نقص ها و کمبودهای نسخه قبلی را برطرف کرده است.
- به عنوان استاندارد اینترنتی RFC 4120 در نظر گرفته شده است.
- در ویندوز از استاندارد اینترنتی کربروس نسخه ۵ به عنوان روش اصلی احراز اصالت کاربران استفاده می‌کند.

مشکلات کربروس نسخه ۴ و نحوه رفع آنها در نسخه ۵

- وابستگی به یک سیستم رمزنگاری خاص (DES)
  - + در نسخه ۵ می‌توان از هر الگوریتم متقارن استفاده کرد. اطلاعات مربوط به نوع الگوریتم و طول کلید به همراه کلید وجود دارند.
- وابستگی به IP
  - + در نسخه ۵ می‌توان از هر نوع آدرس شبکه (مثلا OSI یا IP) استفاده کرد.
- محدود بودن زمان اعتبار بلیط‌ها (مضربی از ۵ دقیقه تا سقف ۲۱ ساعت)
  - + در نسخه ۵ می‌توان ابتدا و انتهای بازه را مشخص کرد.

مشکلات کربروس نسخه ۴ و نحوه رفع آنها در نسخه ۵

- رمزگذاری مضاعف در مرحله ۲ و ۴ (با کلید کارگزار سرویس و کلید کاربر)
  - + در نسخه ۵ از این هزینه اضافی جلوگیری شده است.
- استفاده از مُد غیراستاندارد PCBC در استفاده از DES، که آسیب پذیر است.
  - + در نسخه ۵ از مُد CBC استفاده می شود و قبل از رمز شدن چکیده پیام نیز به آن اضافه می شود.
- امکان حمله دیکشنری برای استخراج گذرواژه و جعل کاربر.
  - + در نسخه ۵ با استفاده از پیش‌احراز اصالت این حمله را سخت‌تر کرده، ولی به طور کامل جلوی آن را نگرفته 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) $AS \rightarrow C$: $Realm_c \| ID_c \| Ticket_{tgs} \| E_{Kc} [K_{c,tgs} \| Times \| Nonce_1 \| Realm_{tgs} \| ID_{tgs}]$

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

(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) $C \rightarrow V$: $Options \| Ticket_v \| Authenticator_c$

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

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

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

کربروس نسخه ۵

- Authentication Service Exchange
  - Realm: دامنه کاربر
  - Options: تقاضای وجود برخی پارامترها در بلیط درخواستی
  - Times: زمان شروع و پایان اعتبار بلیط
  - Nonce: عدد تصادفی برای اطمینان از تازگی پیام دوم
- Client/Server Authentication Exchange
  - Subkey: کلید اختیاری کاربر برای استفاده در نشست جاری. در صورت خالی بودن این فیلد، از $K_{C,V}$ استفاده می‌شود.
  - Seq#: شماره سریال آغازین برای استفاده در پیام‌های ارسالی از کاربر به کارگزار و بالعکس.

پیاده‌سازی‌های موجود

- دانشگاه MIT: اولین پیاده سازی کربروس که هنوز به عنوان مرجع مورد استفاده قرار می‌گیرد.
- Heimdal: پیاده‌سازی انجام شده در خارج آمریکا.
- Active Directory: پیاده‌سازی ارائه شده توسط مایکروسافت که در RFC 1510 آمده است.

برنامه‌های Kerberized

- یکی از روشهای پیاده‌سازی SSO در سازمانها، توسعه برنامه‌های Kerberized است.
  - برنامه‌هایی که قادرند با بلیتهای کربروس کار کنند و بر اساس این بلیتها به کاربران خدمت دهند.
- APIهای متنوعی برای Kerberize نمودن برنامه‌ها وجود دارد.
- مرورگرها نیز می‌توانند با بلیتهای کربروس کار کنند.
  - مناسب برای کاربردهای مبتنی بر وب.

پایان

مرکز امنیت داده و شبکه شریف

http://dnsl.ce.sharif.edu

پست الکترونیکی

amini@sharif.edu