



امنیت داده و شبکه

احراز اصالت کاربر و پروتکل کربروس



فهرست

□ احراز اصالت کاربر

□ معرفی و تاریخچه کربروس

□ دیالوگ‌های ساده احراز اصالت

□ کربروس نسخه ۴

□ کربروس نسخه ۵



احراز اصالت کاربر

- اولین لایه دفاعی در بسیاری از سیستم‌ها
- پیش‌نیاز بسیاری از مکانیزم‌های امنیتی از جمله کنترل دسترسی و حسابرسی

□ **تعریف احراز اصالت کاربر** (طبق RFC 4949)

■ فرآیند واریسی هویت یا شناسه مورد ادعای یک کاربر



احراز اصالت کاربر

□ فرآیند احراز هویت کاربر شامل دو گام است:

■ **گام شناسایی (Identification):** فراهم کردن هویت یا شناسه به سیستم

(به طور مثال: ارائه نام کاربری در بسیاری از سیستم‌ها)

■ **گام واریسی (Verification):** فراهم کردن اطلاعاتی جهت اثبات تعلق

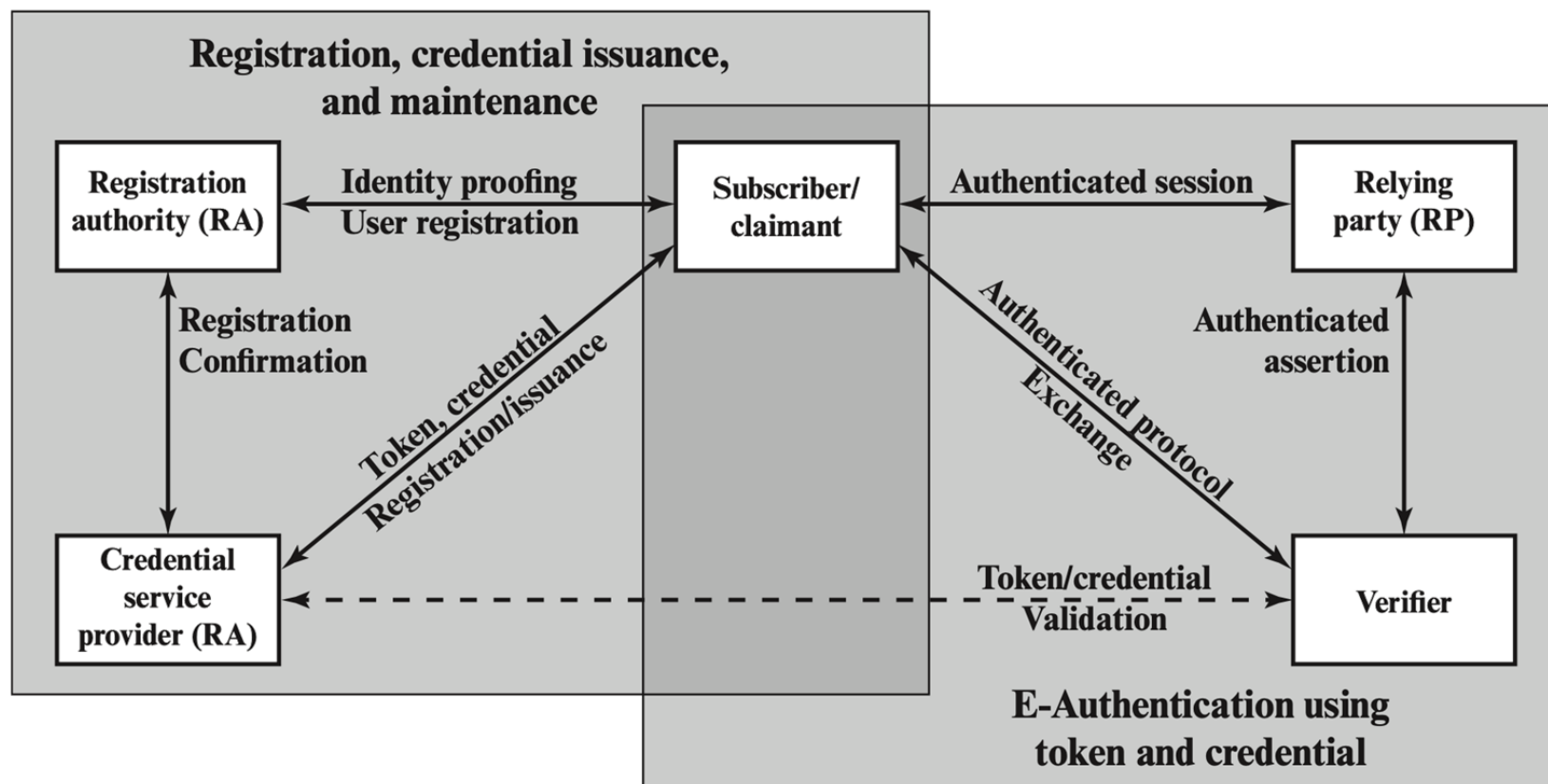
هویت/شناسه ارائه شده به کاربر مدعی (به طور مثال: ارائه گذرواژه جهت

اثبات ادعا)



مدل NIST برای احراز اصالت کاربر (۱)

مدل معماری احراز اصالت کاربر مبتنی بر NIST SP 800-63-2





مدل **NIST** برای احراز اصالت کاربر (۲)

Subscriber/Claimant □

- کاربر که در فرآیند ثبت نام به عنوان مشترک (Subscriber) و در فرآیند احراز اصالت به عنوان مدعی (Claimant) شناخته می شود.

Registration Authority (RA) □

- دریافت درخواست ثبت نام/اشتراک از کاربر
- تایید هویت کاربر و ارسال آن به مولفه CSP

Credential Service Provider (CSP) □

- ارائه اعتبارنامه (credential) به کاربر



مدل NIST برای احراز اصالت کاربر (۳)

□ اعتبارنامه (credential)

- یک ساختار داده شامل هویت کاربر و مجموعه‌ای از ویژگی‌ها (صفات) است که به گونه‌ای به توکن متعلق به یک مشترک/کاربر الحاق می‌شود و می‌تواند برای واریسی به مولفه Verifier ارائه شود.
- توکن یک کلید رمزنگاری و یا یک گذرواژه است که برای احراز یا اثبات هویت مشترک/کاربر به کار می‌رود.
- توکن می‌تواند توسط CSP صادر شود یا مستقیماً توسط خود مشترک/کاربر تولید شود و یا توسط یک شخص ثالث فراهم شود.



مدل NIST برای احراز اصالت کاربر (۴)

Verifier □

- واریسی ادعای مدعی/کاربر از طریق فرآیند احراز اصالت مدعی/کاربر
- اطمینان از اینکه کاربر مدعی همان مشترک ذکر شده در اعتبارنامه است.
- احراز اصالت: اثبات مالکیت توکن و تحت کنترل بودن آن توسط مدعی/کاربر

Relying Party (RP) □

- دریافت اظهارنامه (assertion) شامل شناسه و صفات کاربر از Verifier
- استفاده از اطلاعات احراز شده برای کنترل دسترسی یا مجازشماری



روشهای احراز اصالت کاربر

□ بر اساس دانسته‌های کاربر

خطر افشا یا حدس زدن و یا فراموشی

مانند گذرواژه یا PIN

□ بر اساس داشته‌های کاربر

خطر سرقت یا مفقود شدن

مانند انواع کارت‌های مغناطیسی یا هوشمند، توکن سخت‌افزاری

□ بر اساس مشخصات بیولوژیکی کاربر

خطر خطا در تشخیص و هزینه بالا

مانند اثر انگشت، چهره، شبکیه چشم

□ بر اساس آنچه کاربر انجام می‌دهد


خطر خطا در تشخیص و هزینه بالا

مانند ریتم تایپ کردن، نحوه صحبت کردن، یا نحوه نوشتن با دست



احراز اصالت بر مبنای گذرواژه (۱)

□ طرح ۱: گذرواژه


Pwd = 123456

ID, Pwd



ID	Pwd
myuser	mypass
...	...

□ ضعف امنیتی:

■ لو رفتن گذرواژه (Pwd) در شبکه ارتباطی نا امن

■ حمله تکرار (Replay Attack)



استخراج گذرواژه با شنود روی شبکه

شنود با استفاده از ابزار Wireshark

39	2.450321	213.233.168.3	213.233.168.156	TCP
40	2.450331	213.233.168.156	213.233.168.3	TCP
41	2.450424	213.233.168.156	213.233.168.3	HTTP
42	2.450688	213.233.168.3	213.233.168.156	TCP
43	2.491468	Intel_5b:f3:5e	Broadcast	ARP
44	2.491670	fc80::822::edfa:db2d:8	ff02::1::ffff:1:301	TCP

Source port: stun (3478)			
Destination port: http (80)			
[Stream index: 5]			
Sequence number: 1 (relative sequence number)			
[Next sequence number: 720 (relative sequence number)]			
Acknowledgement number: 1 (relative ack number)			
Header length: 20 bytes			

0230	36 38 63 63 35 34 63 62	62 37 39 39 61 37 31 64	68cc54cb b799a71d
0240	3b 20 50 48 50 53 45 53	53 49 44 3d 31 33 65 35	; PHPSES SID=13e5
0250	62 36 35 33 36 62 30 38	66 32 61 39 33 33 38 36	b6536b08 f2a93386
0260	61 31 33 37 32 66 37 65	64 39 35 39 0d 0a 43 6f	a1372f7e d959..Co
0270	6e 74 65 6e 74 2d 54 79	70 65 3a 20 61 70 70 6c	ntent-Type: appl
0280	69 63 61 74 69 6f 6e 2f	78 2d 77 77 77 2d 66 6f	ication/ x-www-fo
0290	72 6d 2d 75 72 6c 65 6e	63 6f 64 65 64 0d 0a 43	rm-urlencoded..C
02a0	6f 6e 74 65 6e 74 2d 4c	65 6e 67 74 68 3a 20 38	ontent-Length: 8
02b0	30 0d 0a 0d 0a 6c 6f 67	69 6e 5f 75 73 65 72 6e	0....log in_usern
02c0	61 6d 65 3d 6d 5f 61 6d	69 6e 69 26 73 65 63 72	ame=m_am ini&secr
02d0	65 74 6b 65 79 3d 6d 79	70 61 73 73 26 6a 73 5f	etkey=my pass&js
02e0	61 75 74 6f 64 65 74 65	63 74 5f 72 65 73 75 6c	autodete ct_resul
02f0	74 73 3d 31 26 6a 75 73	74 5f 6c 6f 67 67 65 64	ts=1&jus t_logged
0300	5f 69 6e 3d 31		_in=1

Text item 0, 80 bytes	Packets: 338 Displayed: 338 Marked: 0 Dropped: 0
-----------------------	--



احراز اصالت بر مبنای گذرواژه (۲)

طرح ۲: اعمال یک تابع چکیده‌ساز (Hash) روی گذرواژه



ID, Hash(Pwd)



ID	Hash(Pwd)
myuser	7696312368623
...	...

Pwd1 = 123456 → SHA1(Pwd1) = 7c4a8d09 ca3762af 61e59520 943dc264 94f8941b

Pwd2 = 023456 → SHA1(Pwd2) = 393d35d5 521a8dd4 f69a8b98 639f5865 f9c3ff27

□ خواص تابع چکیده‌ساز

■ یک طرفه بودن تابع: با داشتن Hash(Pwd) نمی‌توان Pwd را به دست آورد.

■ با کوچکترین تغییر در ورودی، خروجی کاملاً تغییر می‌کند.

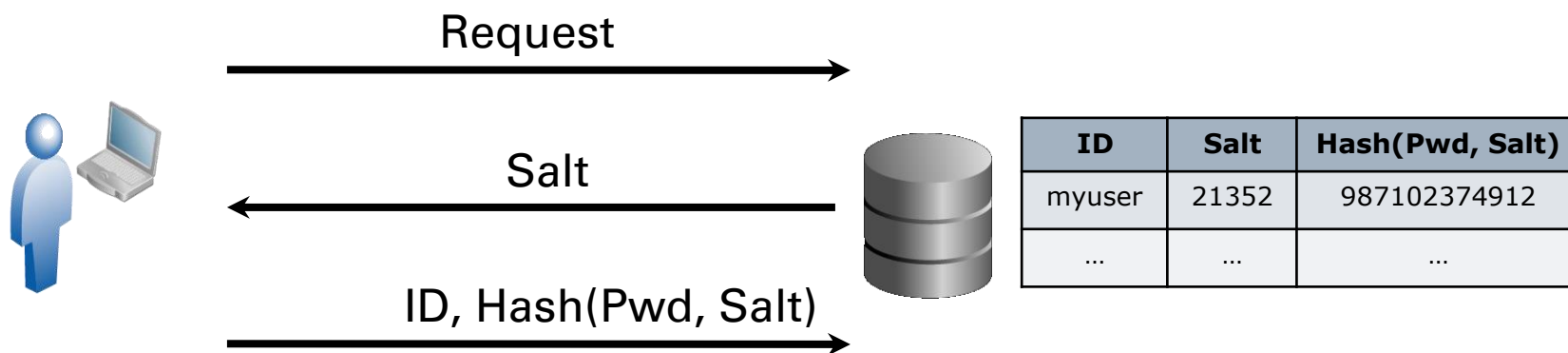
□ ضعف امنیتی:

■ لو رفتن Hash(Pwd)، حمله تکرار



احراز اصالت بر مبنای گذرواژه (۳)

طرح ۳: اعمال یک تابع چکیده‌ساز روی گذرواژه و یک مقدار Salt



□ خصوصیات

■ Salt مقداری تصادفی که به ازای هر شناسه ثابت و ذخیره شده است.

■ عدم تشخیص گذرواژه‌های یکسان

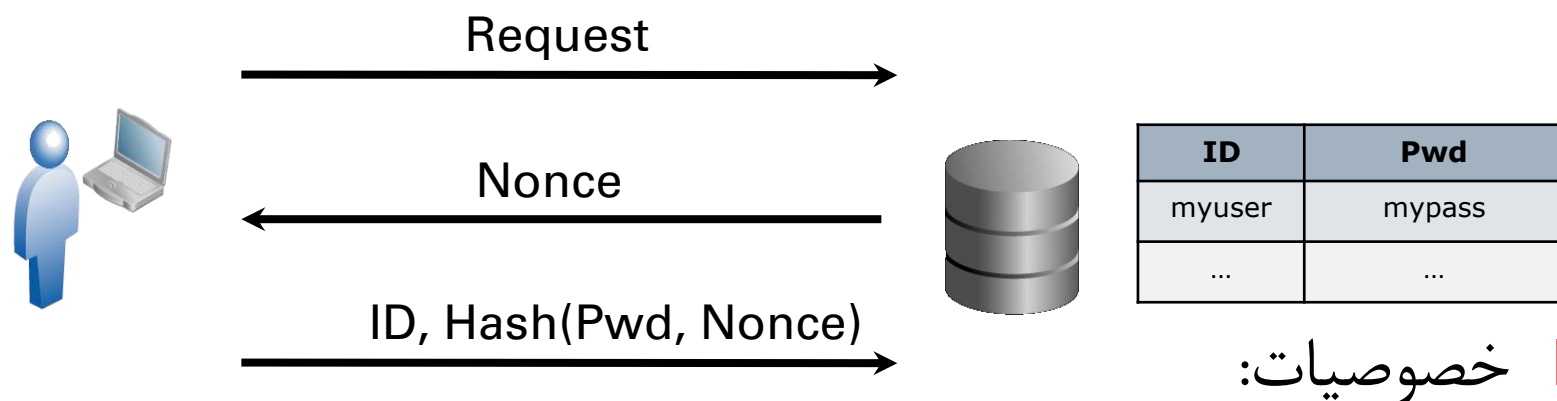
□ ضعف امنیتی

■ لو رفتن Hash(Pwd, Salt)، حمله تکرار



احراز اصالت بر مبنای گذرواژه (۴)

□ طرح ۴: یک پروتکل احراز اصالت مبتنی بر Nonce-handshake



■ مقاوم در برابر حمله تکرار

■ امکان استفاده از رمز متقارن به جای تابع چکیده ساز $E_{KPwd}(Nonce)$

□ ضعف امنیتی:

■ وابستگی به تنها یک عامل امنیتی

■ حمله حدس گذرواژه (Password Guessing Attack, Dictionary Attack)



حمله حدس گذرواژه

□ مهاجم با شنود خط ارتباطی (یک بار) می داند که

Nonce = 99887766

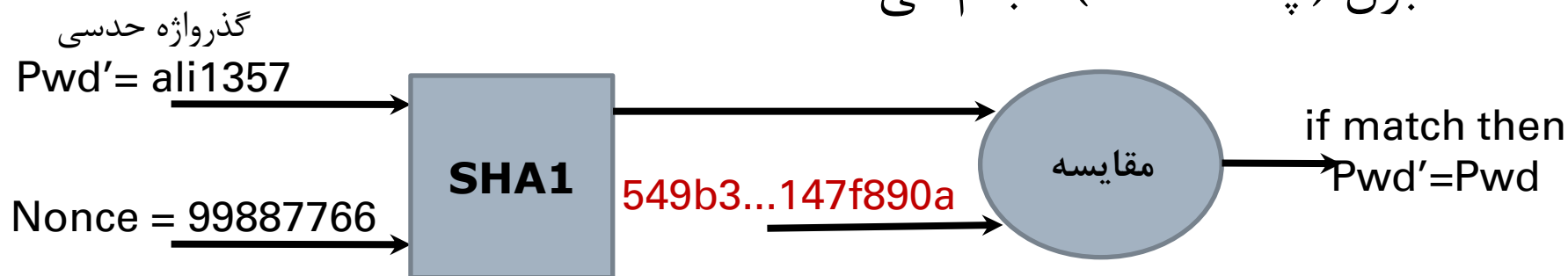
SHA1(Pwd, Nonce)=549b386d830fcbcf74fc5049fee40543147f890a

□ مهاجم به یک مخزن (Dictionary) از گذرواژه‌های محتمل دسترسی دارد.

■ فرهنگ لغات انگلیسی و فارسی

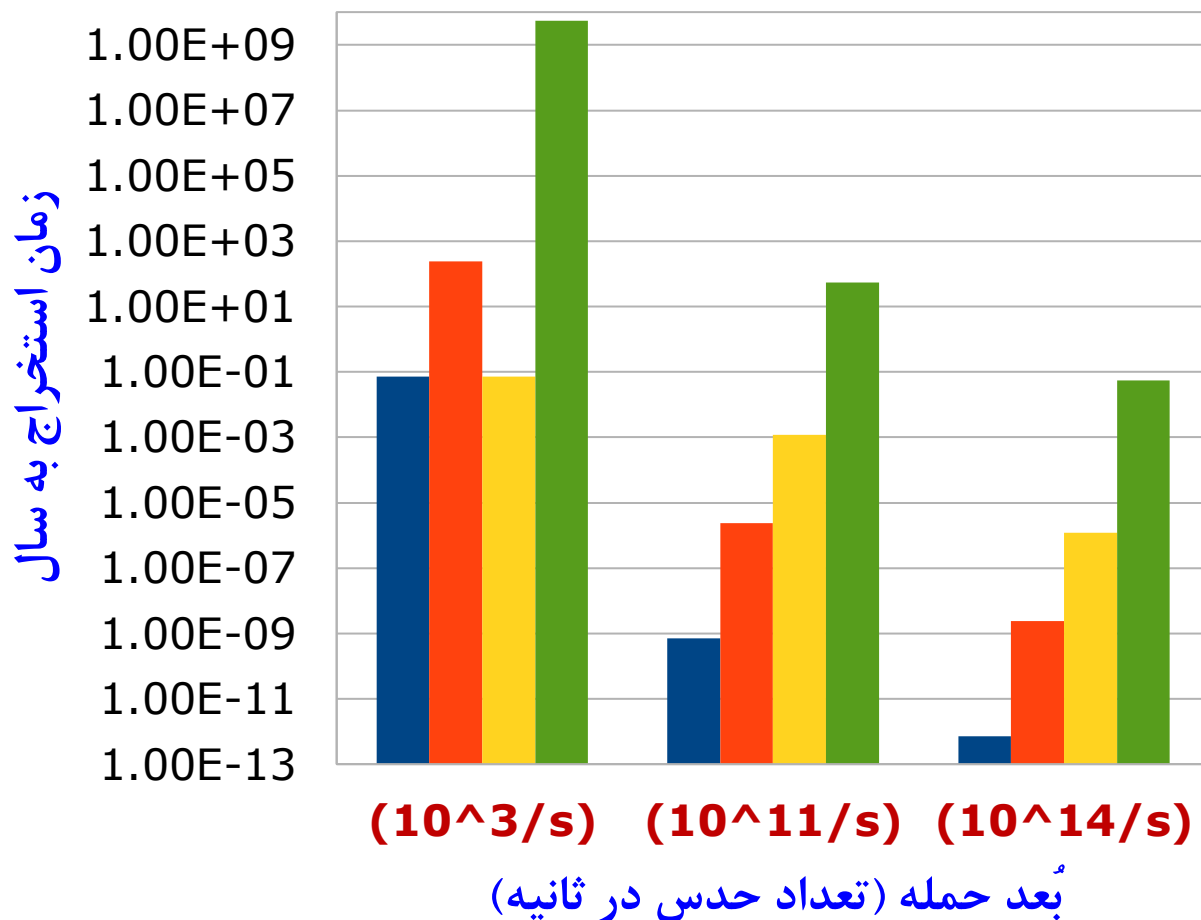
■ اعداد رایج در گذرواژه‌ها ۱۲۳۴۵۶، ۱۳۰۱ تا ۱۳۹۵، ۱۹۰۰ تا ۲۰۱۶

□ به ازای هر گذرواژه محتمل Pwd'، مهاجم محاسبات زیر را در زمان قابل قبول (چند ساعت) انجام می‌دهد.





پیچیدگی زمانی حدس گذرواژه



زمان	معادل سال
ثانیه	3.171E-08
دقیقه	1.903E-06
ساعت	0.0001142
روز	0.0027397
هفته	0.0191781
ماه	0.0821918
سال	1

- 6 Letters
- 6 Letters + 1 Symbol
- 10 Letters
- 10 Letters + 1 Symbol

منبع: <http://www.itworld.com/article/2832596/security/how-many-seconds-would-it-take-to-break-your-password-.html>



تاثیر توان محاسباتی مهاجم در اجرای حمله

□ گذرواژه ۸ حرفی شامل حروف کوچک، بزرگ و اعداد نیز ضعیف است اگر

■ مهاجم به **رایانه‌ای با قدرت ده میلیون برابر** دسترسی داشته باشد.

□ دسترسی احتمالی سازمان‌های اطلاعاتی/جاسوسی

□ پیشرفت تکنولوژی در آینده و دسترسی کاربران عادی

■ مهاجم **میلیون‌ها رایانه را کنترل و حمله را به صورت توزیع شده** اجرا کند.

□ با استفاده از یک شبکه بات امکان پذیر است.

□ **یادآوری:** گذرواژه باید به اندازه کافی پیچیده باشد:

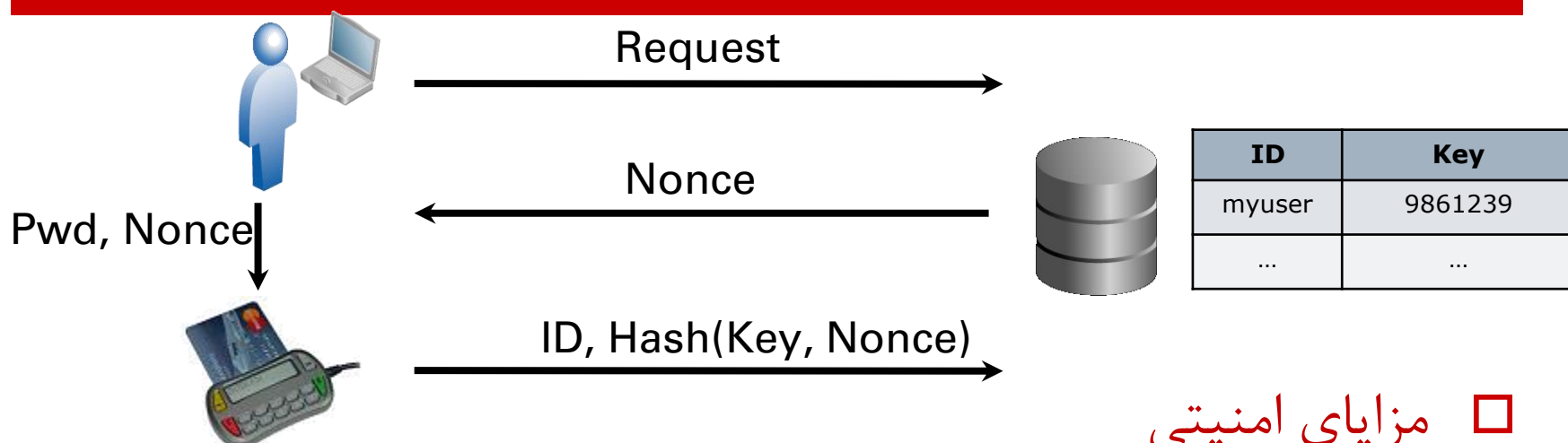
□ طول مناسب

□ حروف کوچک، بزرگ، اعداد، و

□ سایر نمادهای صفحه کلید (. * + - _ / % ^ # @ ! ...)



احراز اصالت قوی مبتنی بر کارت هوشمند

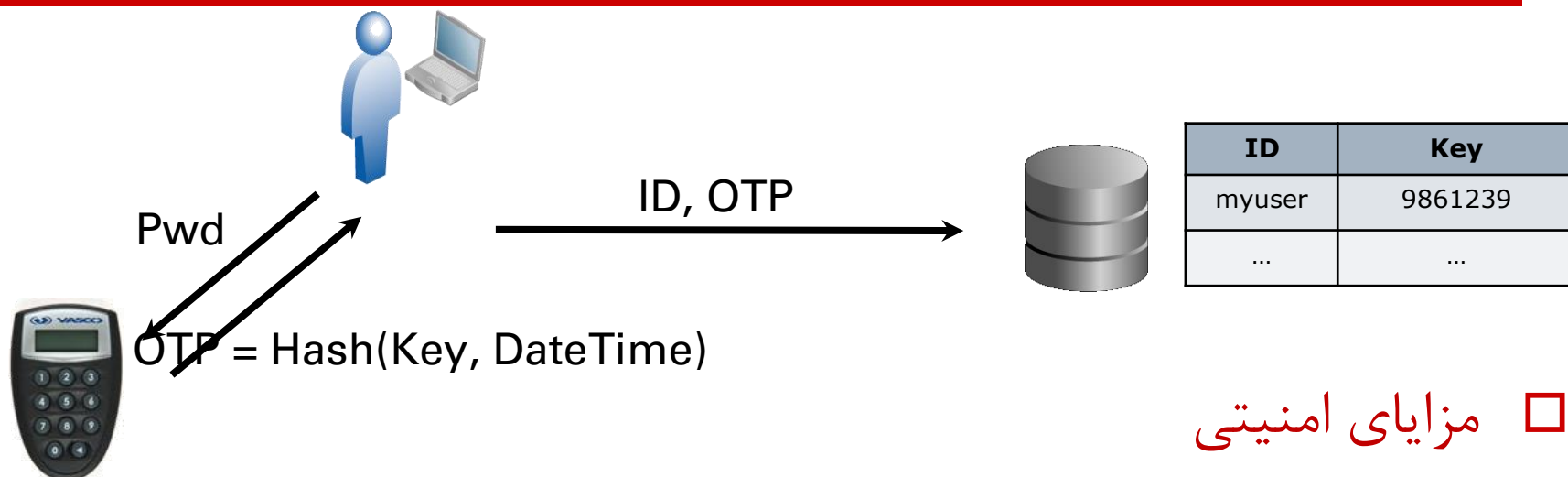


□ مزایای امنیتی

- کلید امنیتی Key با طول زیاد در کارت هوشمند ذخیره شده و هرگز از آن خارج نمی‌شود.
- کارت هوشمند تنها با وارد کردن Pwd فعال می‌شود (مقاوم در برابر سرقت)
- توکن‌های USB هم پروتکل و خواص امنیتی مشابهی دارند.
- راه‌حل‌های مبتنی بر PKI و امضای دیجیتال، انکار یک ارتباط توسط کاربر را ناممکن می‌کنند.



احراز اصالت قوی مبتنی بر OTP



مزایای امنیتی

- کلید امنیتی Key در دستگاه (OTP) ذخیره شده است و هرگز از آن خارج نمی‌شود.
- OTP تنها با وارد کردن Pwd فعال می‌شود و یک خروجی نمایش می‌دهد که در یک بازه زمانی کوتاهی (مثلاً ۱ دقیقه) معتبر است.
- کارگزار برای زمانهای متفاوت (تا یک دقیقه) قبل را محاسبه و با OTP مقایسه می‌کند.
- ساعت ها باید همزمان باشند (با در نظر گرفتن تاخیر خط ارتباطی).



فهرست

□ احراز اصالت کاربر

□ معرفی و تاریخچه کربروس

□ دیالوگ‌های ساده احراز اصالت

□ کربروس نسخه ۴

□ کربروس نسخه ۵



معرفی کربروس (Kerberos یا Cerberus)

□ **Kerberos (Κέρβερος):** بر گرفته از اسطوره یونانی

■ نام سگی سه سر که محافظ دروازه های عالم مردگان بود؛ نمی گذاشت زندگان مزاحم ارواح شده و ارواح از عالم مردگان خارج شوند.

□ سرها نماد:



■ احراز اصالت (Authentication)

■ مجازشماری (Authorization)

■ حسابرسی (Accounting)



تاریخچه کربروس - 1

□ 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 محدودیتهای و اشکالات امنیتی فراوان



تاریخچه کربروس - 2

□ نسخه ۵ در ۱۹۹۳ به طور رسمی منتشر شد (RFC 1510).

□ تا سال ۲۰۰۰، رمزنگاری در آمریکا «سلاح» محسوب می شد.

■ 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 - 3

- 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 - 4

□ 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 - 5

□ 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 پیاده سازی 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 دسترس پذیری کارگزار احراز اصالت کربروس با پشتیبانی از خدمت رسانی توزیع شده و کارگزار پشتیبان.

□ پنهانی (Transparency)


- کاربران باید سیستم را همانند یک سیستم ساده مبتنی بر گذرواژه ببینند.

□ مقیاس پذیری (Scalability)

- قابلیت کار با تعداد زیادی ایستگاه کاری و کارگزار با پشتیبانی از ساختاری پیمانه‌ای و توزیع شده.



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.

□ کاربر قصد دریافت خدمات از کارگزار سرویس را دارد.

■ **C→AS:** $ID_C \parallel P_C \parallel ID_V$

■ **AS→C:** Ticket

■ **C→V:** $ID_C \parallel$ Ticket

■ Ticket = $E(K_V, [ID_C \parallel AD_C \parallel ID_V])$

C = کلاینت

AS = کارگزار احراز اصالت

V = کارگزار سرویس

P_C = گذرواژه کاربر پشت کلاینت موردنظر

AD_C = آدرس شبکه کلاینت

K_V = کلید اصلی بین AS و V



بلیط

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



بررسی دیالوگ

□ چرا آدرس کارفرما (Client) در بلیط ذکر می شود؟

■ در غیر این صورت هر شخصی که بلیط را از طریق شنود به دست آورد نیز می تواند از امکانات استفاده کند. اما اکنون تنها خدمات به آدرس ذکر شده در بلیط ارائه می شود.

□ مشکل جعل آدرس

□ چرا شناسه کارفرما ID_C در گام سوم به صورت رمز نشده ارسال می شود؟

■ زیرا این اطلاعات به صورت رمز شده در بلیط وجود دارد.

■ اگر شناسه با بلیط مطابقت نداشته باشد خدمات ارائه نمی شوند.



مشکلات دیالوگ ساده احراز اصالت

□ ناامنی

- ارسال کلمه عبور بدون رمزگذاری
- امکان حمله تکرار (با شنود، جعل شناسه و جعل آدرس کاربر)

□ ناکارایی

- لزوم احراز اصالت کاربر برای دریافت بلیط جدید برای هر خدمت



ارتقای امنیت-دیالوگ 1

□ 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 می شود.

□ بلیط های اعطاء خدمات توسط **TGS** صادر می شوند.

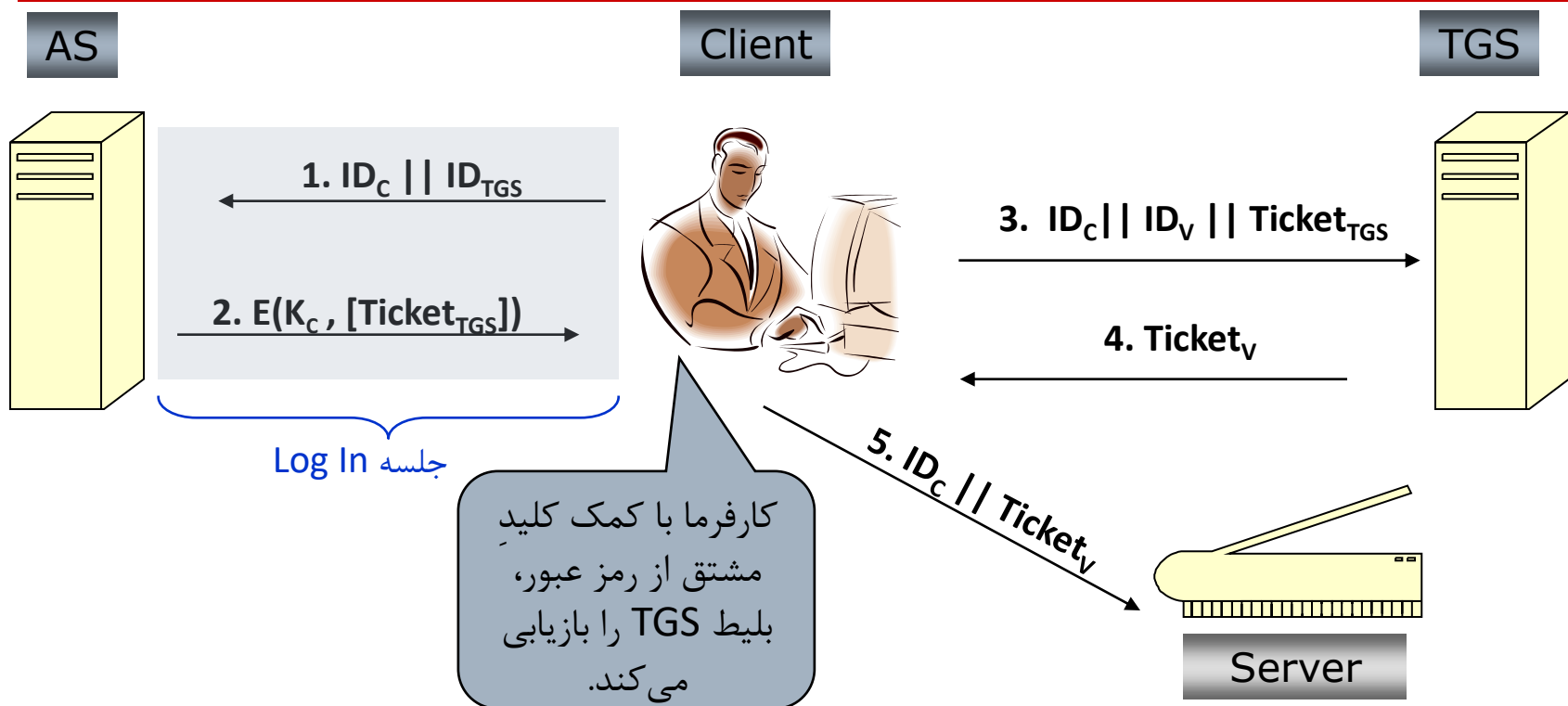
■ بلیط “اعطاء خدمات” **service-granting ticket**

□ 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-دیالوگ 1



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

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



ارتقای امنیت-دیالوگ 1

□ 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. $C \rightarrow AS: ID_C \parallel ID_{TGS}$
2. $AS \rightarrow C: E(K_C, Ticket_{TGS})$
3. $C \rightarrow TGS: ID_C \parallel ID_V \parallel Ticket_{TGS}$
4. $TGS \rightarrow C: Ticket_V$
5. $C \rightarrow V: ID_C \parallel Ticket_V$



محتوي بلیط ها

□ بلیط اعطای بلیط:

$$\text{Ticket}_{\text{TGS}} = E(K_{\text{TGS}}, [ID_C \parallel \text{Addr}_C \parallel ID_{\text{TGS}} \parallel \text{Timestamp}_1 \parallel \text{Lifetime}_1])$$

□ بلیط اعطای خدمات:

$$\text{Ticket}_V = E(K_V, [ID_C \parallel \text{Addr}_C \parallel ID_V \parallel \text{Timestamp}_2 \parallel \text{Lifetime}_2])$$



ویژگی های دیالوگ 1

- دو بلیط صادر شده ساختار مشابهی دارند. در اساس به دنبال هدف واحدی هستند.
- رمزنگاری $\text{Ticket}_{\text{TGS}}$ جهت احراز اصالت
 - تنها کارفرما می تواند به بلیط رمز شده دسترسی پیدا کند.
- رمز نمودن محتوای بلیطها صحت را فراهم می کند.
- استفاده از مُهر زمانی (Timestamp) در بلیطها، آنها را برای یک بازه زمانی تعریف شده قابل استفاده مجدد می کند.
- هنوز از آدرس شبکه برای احراز اصالت بهره می گیرد.
 - چندان جالب نیست زیرا آدرس شبکه، قابل جعل (Spoof) است.
 - با این حال، درجه ای از امنیت مهیا می شود.



نقاط ضعف دیالوگ 1

□ 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

■ امکان رسیدن درخواست‌ها به یک کارگزار جعلی و غیرمجاز به جای کارگزار اصلی V.



فهرست

□ مقدمه و نیازمندی‌ها

□ دیالوگ‌های ساده احراز اصالت

□ **کربروس نسخه ۴**

□ کربروس نسخه ۵



کربروس نسخه 4

□ 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

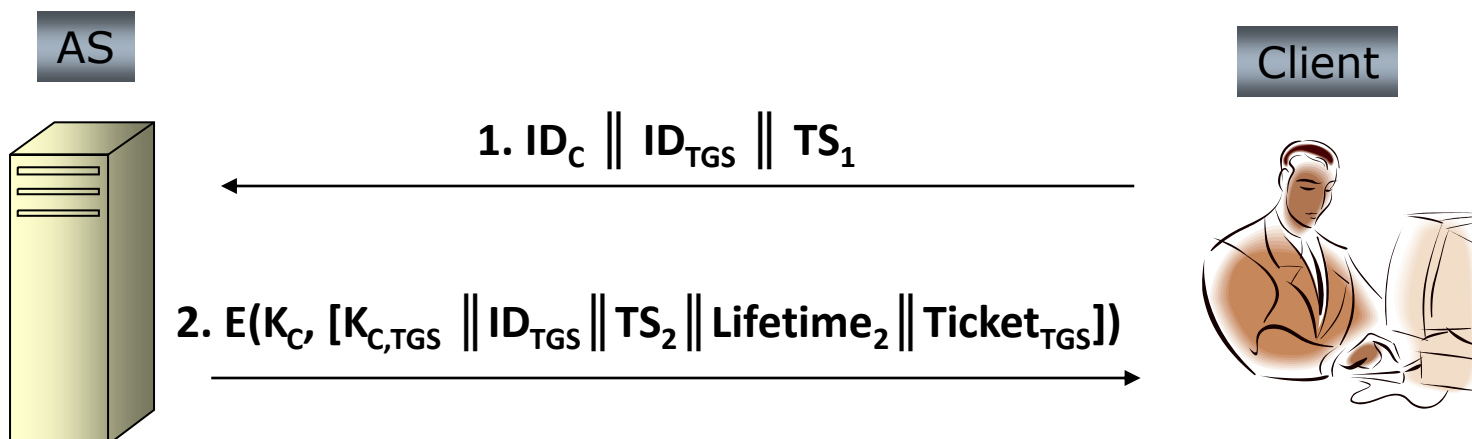
□ یک نیاز جدید: کارگزار یا TGS باید اطمینان حاصل نمایند که کاربرِ بلیط، همان کسی است که بلیط برای او صادر شده.

□ مفهوم جدیدی به نام احراز کننده (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

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



بدست آوردن بلیط اعطاء بلیط

□ نتایج این مرحله برای کارفرما

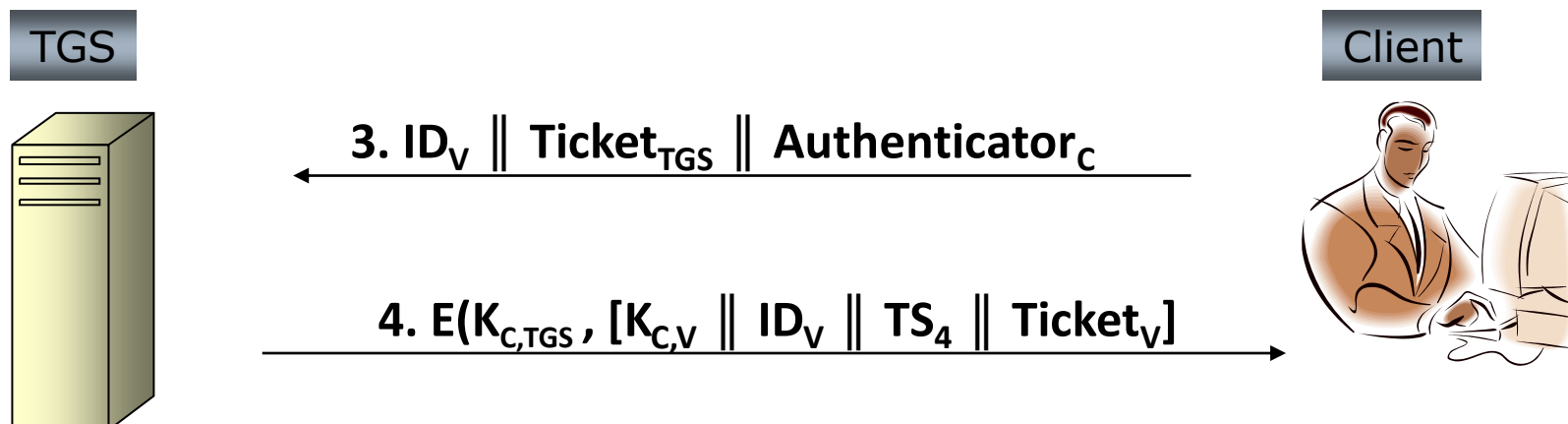
■ بدست آوردن امن بلیط "اعطاء بلیط" از AS

■ بدست آوردن زمان انقضای بلیط (TS_2)

■ بدست آوردن کلید جلسه امن بین کارفرما و TGS



بدست آوردن بلیط اعطاء خدمات



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

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

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



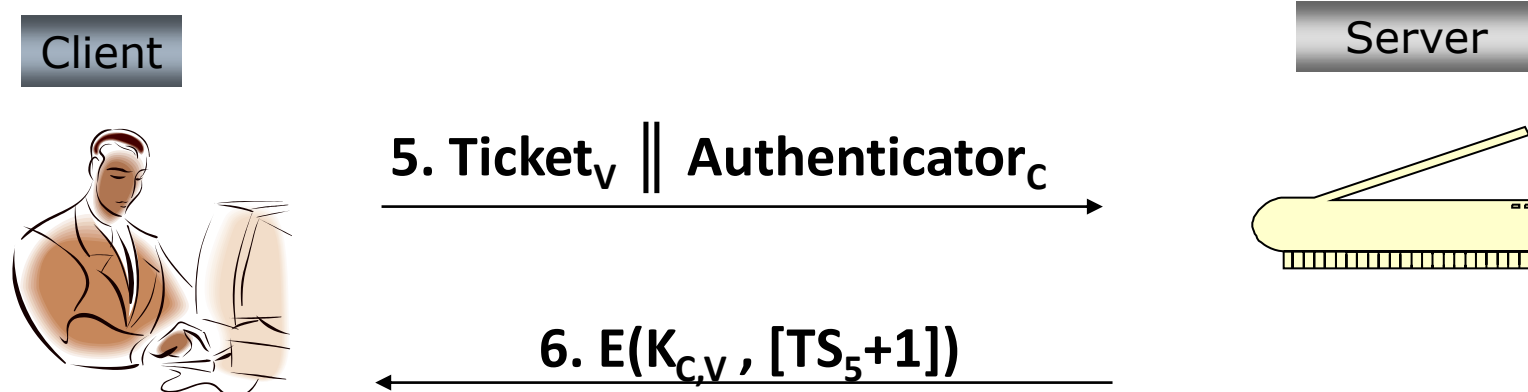
بدست آوردن بلیط اعطاء خدمات

□ نتایج این مرحله برای کارفرما

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



دستیابی به خدمات کارگزار



$$\text{Ticket}_v = E(K_v, [K_{c,v} \parallel ID_c \parallel \text{Addr}_c \parallel ID_v \parallel TS_4 \parallel \text{Lifetime}_4])$$

$$\text{Authenticator}_c = E(K_{c,v}, [ID_c \parallel \text{Addr}_c \parallel TS_5])$$



دستیابی به خدمات کارگزار

□ نتایج این مرحله برای کارفرما

■ احراز اصالت کارگزار در گام ششم با برگرداندن پیغام رمز شده

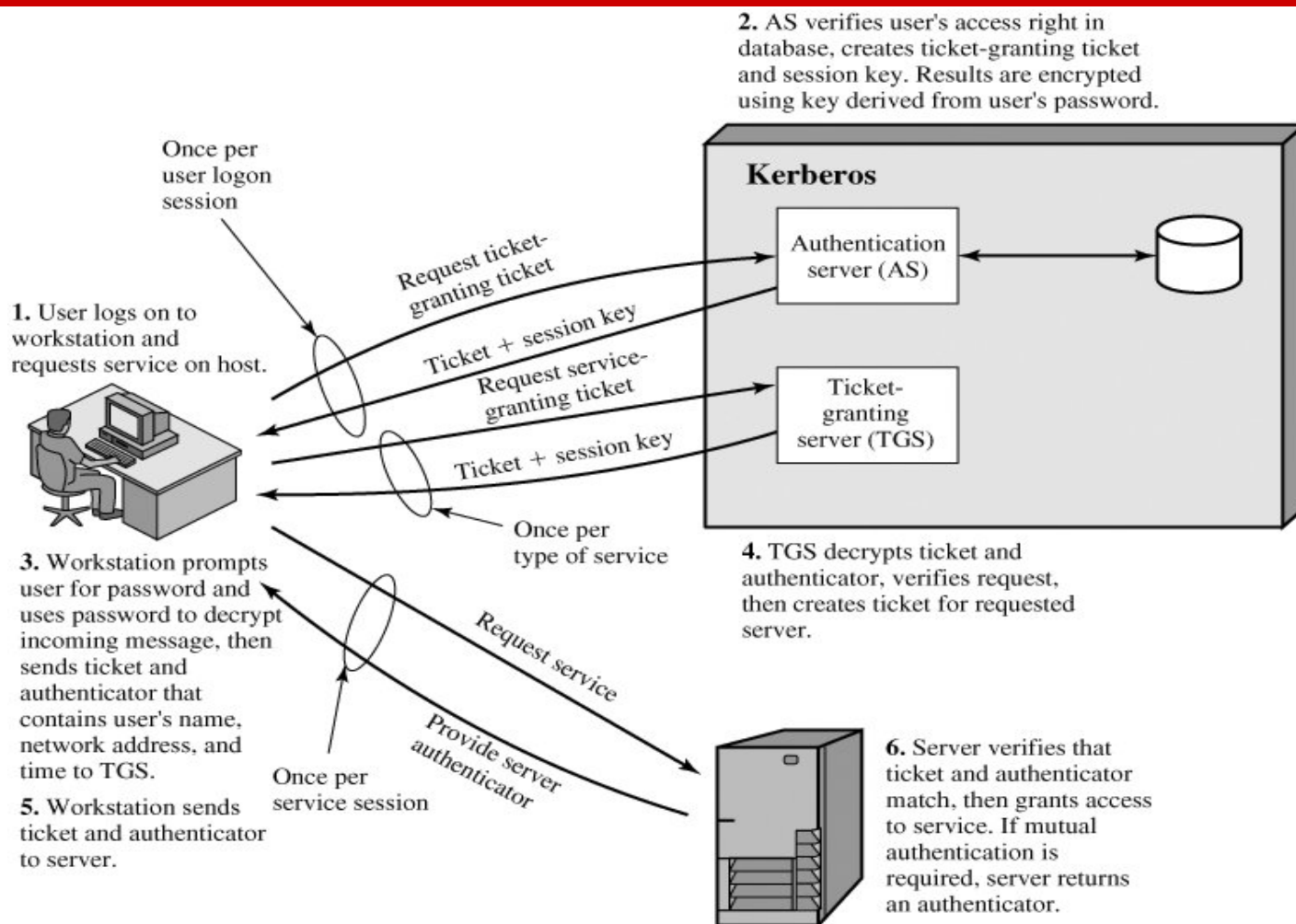
■ جلوگیری از بروز حمله تکرار



کربروس نسخه 4: شمای کلی

(a) Authentication Service Exchange: to obtain ticket-granting ticket
<p>(1) $C \rightarrow AS: ID_c \parallel ID_{tgs} \parallel TS_1$</p> <p>(2) $AS \rightarrow C: E_{K_c}[K_{c,tgs} \parallel ID_{tgs} \parallel TS_2 \parallel Lifetime_2 \parallel Ticket_{tgs}]$</p> <p style="text-align: center;">$Ticket_{tgs} = E_{K_{tgs}}[K_{c,tgs} \parallel ID_C \parallel AD_C \parallel ID_{tgs} \parallel TS_2 \parallel Lifetime_2]$</p>
(b) Ticket-Granting Service Exchange: to obtain service-granting ticket
<p>(3) $C \rightarrow TGS: ID_v \parallel Ticket_{tgs} \parallel Authenticator_c$</p> <p>(4) $TGS \rightarrow C: E_{K_{c,tgs}}[K_{c,v} \parallel ID_v \parallel TS_4 \parallel Ticket_v]$</p> <p style="text-align: center;">$Ticket_{tgs} = E_{K_{tgs}}[K_{c,tgs} \parallel ID_C \parallel AD_C \parallel ID_{tgs} \parallel TS_2 \parallel Lifetime_2]$</p> <p style="text-align: center;">$Ticket_v = E_{K_v}[K_{c,v} \parallel ID_C \parallel AD_C \parallel ID_v \parallel TS_4 \parallel Lifetime_4]$</p> <p style="text-align: center;">$Authenticator_c = E_{K_{tgs}}[ID_C \parallel AD_C \parallel TS_3]$</p>
(c) Client/Server Authentication Exchange: to obtain service
<p>(5) $C \rightarrow V: Ticket_v \parallel Authenticator_c$</p> <p>(6) $V \rightarrow C: E_{K_{c,v}}[TS_5 + 1]$ (for mutual authentication)</p> <p style="text-align: center;">$Ticket_v = E_{K_v}[K_{c,v} \parallel ID_C \parallel AD_C \parallel ID_v \parallel TS_4 \parallel Lifetime_4]$</p> <p style="text-align: center;">$Authenticator_c = E_{K_{c,v}}[ID_C \parallel AD_C \parallel TS_5]$</p>

کربروس نسخه 4: 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 پایگاه داده‌های خود دارد.
- کارگزار کربروس با هر کارگزار برنامه کاربردی کلیدی مخفی به اشتراک گذاشته است.
- معمولاً هر دامنه معادل یک حوزه مدیریتی است.



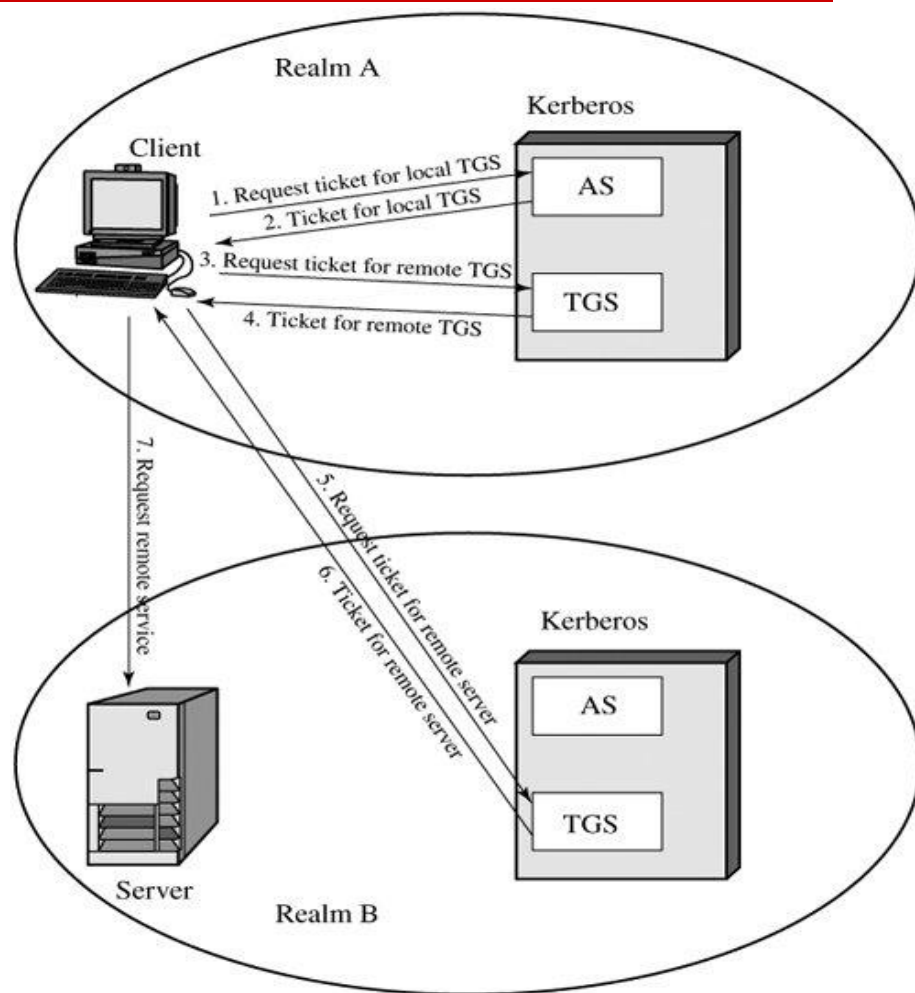
تعامل بین دامنه‌ای

- 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

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





فهرست

□ مقدمه و نیازمندی‌ها

□ دیالوگ‌های ساده احراز اصالت

□ کربروس نسخه ۴

□ کربروس نسخه ۵



کربروس نسخه 5

□ 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 می کند.



مشکلات کربروس نسخه 4 و نحوه رفع آنها در نسخه 5

□ وابستگی به یک سیستم رمزنگاری خاص (DES)

■ + در نسخه ۵ می توان 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 می توان 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 می توان ابتدا و انتهای بازه را مشخص کرد.



مشکلات کرپروس نسخه 4 و نحوه رفع آنها در نسخه 5

□ 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.



مشکلات کرپروس نسخه 4 و نحوه رفع آنها در نسخه 5

□ امکان حمله دیکشنری برای استخراج گذرواژه و جعل کاربر.

■ + در نسخه ۵ با استفاده از **پیش احراز اصالت** این حمله را سخت تر کرده، ولی به طور کامل جلوی آن را نگرفته است.

■ برای پیش احراز اصالت، AS قبل از ارسال بلیط TGS، هویت کاربر را با روشی احراز می نماید. به طور مثال کاربر باید رمز شده زمان جاری و نسخه جاری الگوریتم را با کلید حاصل از گذرواژه، رمز و ارسال نماید.

■ از کارت هوشمند، توکن و یا روشهای دیگر نیز برای پیش احراز اصالت می توان استفاده کرد و روش خاصی دیکته نشده است.



کربروس نسخه 5: شمای کلی

(a) Authentication Service Exchange: to obtain ticket-granting ticket	
(1) $C \rightarrow AS$:	$Options \parallel ID_c \parallel Realm_c \parallel ID_{tgs} \parallel Times \parallel Nonce_1$
(2) $AS \rightarrow C$:	$Realm_c \parallel ID_c \parallel Ticket_{tgs} \parallel E_{K_c} [K_{c,tgs} \parallel Times \parallel Nonce_1 \parallel Realm_{tgs} \parallel ID_{tgs}]$
	$Ticket_{tgs} = E_{K_{tgs}} [Flags \parallel K_{c,tgs} \parallel Realm_c \parallel ID_c \parallel AD_c \parallel Times]$
(b) Ticket-Granting Service Exchange: to obtain service-granting ticket	
(3) $C \rightarrow TGS$:	$Options \parallel ID_v \parallel Times \parallel Nonce_2 \parallel Ticket_{tgs} \parallel Authenticator_c$
(4) $TGS \rightarrow C$:	$Realm_c \parallel ID_c \parallel Ticket_v \parallel E_{K_{c,tgs}} [K_{c,v} \parallel Times \parallel Nonce_2 \parallel Realm_v \parallel ID_v]$
	$Ticket_{tgs} = E_{K_{tgs}} [Flags \parallel K_{c,tgs} \parallel Realm_c \parallel ID_c \parallel AD_c \parallel Times]$
	$Ticket_v = E_{K_v} [Flags \parallel K_{c,v} \parallel Realm_c \parallel ID_c \parallel AD_c \parallel Times]$
	$Authenticator_c = E_{K_{c,tgs}} [ID_c \parallel Realm_c \parallel TS_1]$
(c) Client/Server Authentication Exchange: to obtain service	
(5) $C \rightarrow V$:	$Options \parallel Ticket_v \parallel Authenticator_c$
(6) $V \rightarrow C$:	$E_{K_{c,v}} [TS_2 \parallel Subkey \parallel Seq\#]$
	$Ticket_v = E_{K_v} [Flags \parallel K_{c,v} \parallel Realm_c \parallel ID_c \parallel AD_c \parallel Times]$
	$Authenticator_c = E_{K_{c,v}} [ID_c \parallel Realm_c \parallel TS_2 \parallel Subkey \parallel Seq\#]$



کربروس نسخه 5

Authentication Service Exchange ☐

- Realm: دامنه کاربر
- Options: تقاضای وجود برخی پارامترها در بلیط درخواستی (با استفاده از flagها). یکی از انواع flagها مربوط به پیش‌احراز اصالت است.
- 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 پیاده‌سازی 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
