Subject:
Year. Month. Date.

- سرویس امنیتی :

هدف مقابله با حملات امنیتی است.

تعریف ← RFC2828 : یک سرویس ارتباطی با پردازشی برای حفاظت از منابع سیستم.

در استاندارد X800 ۶ سرویس داریم :

- تصدیق اصالت : اطمینان از تطبیق ادعا با فرد ارسال کننده.
- کنترل دسترسی : جلوگیری از دسترسی غیرمجاز
- محرمانگی داده : جلوگیری از افشای اطلاعات
- صحت داده : ارسال و دریافت به درستی صورت گرفته
- عدم انکار : ارسال کننده و دریافت کننده نتواند ارسال یا دریافت را انکار کند.
- دسترس پذیری : در دسترس بودن منابع

برای تهیه این سرویس ها تکنیک های متفاوتی داریم اما مثلاً رمزنگاری جزء تکنیک های بسیار رایج است.

در اسلایدها یک مدل امنیت برای شبکه ارائه شده است که طی آن یک پیام ابتدا رمز می شود و سپس در طول کانال شبکه تبادل می شود. در یک سیستم اطلاعاتی برای بررسی نیازمندی های امنیتی به یک gatekeeper نیاز داریم مثلاً چک کردن یوزر و پسورد کاربران باشد...

۹۶/۷/۴ *

- رمزنگاری متقارن :

یکی از روش های حفاظت از دارایی ما یعنی data می باشد. رمزنگاری یعنی نفوذ هم اگر پروتکل را بداند نداند داده چه است.

plain text → پیام اصلی و خام (خوانا و بدون تغییر)

cipher text → پیامی که با تغییرات کد شده است و خوانا نیست

cipher → الگوریتمی که برای رمز کردن از خام به رمز شده استفاده می شود.

Subject:
Year. Month. Date.

در الگوریتم رمزنگاری از یک داده‌ی کوچک برای رمزنگاری استفاده می‌شود

Key → کلید

encipher → تبدیل متن خام به متن رمزشده به صورت

decipher → بازیابی دوباره‌ی متن خام از متن رمزشده مرتبط

cryptography → مطالعه‌ی اصول و روش‌های رمزنگاری

cryptanalysis → مطالعه‌ی روش‌ها و اصولی برای بازیابی متن اصلی بدون داشتن کلید رمز

cryptology → cryptography + cryptanalysis

متقارن ← چون دو طرف گیرنده و فرستنده دارای یک کلید یکسان هستند تابع رمزنگاری هم معکوس‌پذیر است به گونه‌ای که حتماً نیاز به کلید باشد تا تابع را به دست آورد.

نیازمندی‌ها ← الگوریتم رمزنگاری قوی

کلید سری که تنها فرستنده و گیرنده از آن آگاه باشند

در رمزنگاری فرض بر این است که الگوریتم را همه می‌دانند تا طرف مقابل هم بتواند معکوس کند بنابراین برای توزیع کلید نیاز به کانال امن داریم.

مثال: A می‌خواهد اموال ارزشمند قیمتی را برای همراهانش به B بفرستد. هر کدام یک قفل خودش و کلیدش را دارد. باید سه بار سرود و برگردد: A قفل کند، B قفل کند، A باز کند، B قفل خودش را باز کند. البته man in the middle ممکن است، یعنی باریانت از A، قفل خودش را برای B به صندوق بزند به A برگرداند و A باز می‌کند. حالا نقل قفل پیشینی روی صندوق و او دسترسی دارد. می‌تواند برای اینکه مشکوک نشوند برای B هم نقش A را بازی کند.

برای تقسیم‌بندی الگوریتم‌های رمزنگاری می‌توان براساس عملیات انجام شده، تعداد کلیدهای مورد نیاز، روش پردازش متن.

- عملیات انجام شده ← جایگزینی / ضرب / جمع
- تعداد کلیدهای مورد نیاز ← یک کلید عمومی / دو کلید عمومی یا خصوصی
- پردازش متن پیام ← بلوکی ← داده‌ها جمع می‌شوند و بلوک بلوک رمز می‌شوند
  جریان داده‌ای ← با جریانی رندوم بیت به بیت ⊕ XOR
  جریان رندوم با یک کلید مشترک است.

Subject:
Year. Month. Date.

- انواع حملات تحلیل رمزنگاری:

به عبارت یک سطح بندی است. هدف شکستن متن رمز شده است و یا به دست آوردن کلید.

هدف: به دست آوردن بیت های کلید. فرض بر این است الگوریتم را بدهد و بداند سطح به اسکلات

← مهاجم وابسته است.

← Ciphertext only: تحلیلگر تنها متن رمز شده را دارد.

← Known plain text: تحلیلگر چند نمونه از متن اصلی و متن رمز شده ی متناظر را دارد.

← Chosen plain text: تحلیلگر می تواند الگوریتم رمز را به دست آورد و روی متن های اصلی اعمال کند و متن

رمز شده آنها را به دست آورد و با نمونه گیری به کمک آن کلید را استخراج کند.

← Chosen ciphertext: تحلیلگر متن رمز شده را به الگوریتم بدهد و متن خام دریافت کند.

← Chosen text: تحلیلگر هم می تواند رمز کند و هم می تواند رمز را بشکند.

• Brute Force Attack:

ابتدایی ترین حالت این است که تمام حالات را جستجو کنیم. ولی خب خیلی زمان می خواهد!

می توان با افزایش موجودی فضای حالت را کاهش داد.

* 96/7/11

- امنیت مطلق ← مستقل از قدرت مهاجم، با فرض متن اصلی از روی متن رمز شده به دست نیاید.

← عملی نیست.

- امنیت محاسباتی ← هزینه ی رمزگشایی بیشتر از ارزش اطلاعات و زمان مورد نیاز برای رمزگشایی

بیشتر از طول عمر مفید اطلاعات باشد.

• رمزهای کلاسیک: مثلاً پاپیروس را در چوب می پیچیدند. روی سر خالکوبی کردند ← البته این صرفاً

پنهان سازی داده است و نه رمزنگاری. یه چیزی هم داریم به نام کانال نهانی که داده را در آن برای انتقال

مخفی می کنیم. این دسته از رمزها به صورت دستی انجام می شدند. این رمزها دو روش کلی دارند:

← جایگشتی: با یک جایگشتی حروف را درهم بریزیم و جابه جا کنیم.

← جانشینی: حرف ها جانشین همدیگر می شوند. مثل رمز سزار (یک الفبایی) ← همه ی حروف پایه کلید

(چند الفبایی) ← همه حروف پایه کلید

Subject:
Year. Month. Date.

• جدول Vigenere ← یک ماتریس ۲۶×۲۶ داریم. حروف متوالی کلید ← سطر و حروف متوالی متن ← ستون

• ماشین enigma ← در جنگ جهانی: به سری چرخدنده و روتورها تسلسل‌های متوالی اعمال می‌دهند ایده‌ای شد تا عمل ساده‌ی درهم‌ریختگی را چندین بار تکرار کنند و کمی پیچیده‌تر کنند.

• جایگشت ستونی ← سطری بنویسیم و ستونی بخوانیم

← ایده‌های تحلیل رمز کلاسیک ←
- فراوانی حروف
- فراوانی ترکیبات حروف
- نظم موجود در زبان
- تست Kasiski
- حملات brute Force

• رمزنگاری مدرن: علاوه بر روش‌های کلاسیک از عملیات‌های مختلف هم استفاده می‌کنیم

← تابع رمزنگاری کامل: به ازای یک کلید کاملاً تصادفی برای رمز داشته باشیم. طول این کلید باید حداقل = اندازه‌ی متن خام باشد. در این صورت: $P(M|C) = P(M)$

عملاً ممکن نیست زیرا حجم بالایی باید کلید تولید شود و اگر بتوان کانال امنی برای کلید یافت، از همان کانال داده‌ی اصلی را هم می‌توان منتقل کرد.

← رمزنگاری متقارن: یک کلید مشترک بین دو طرف وجود دارد.

رمزهای قطعه‌ای ← متن به قطعاتی تقسیم شده و قطعات رمز می‌شوند. در طی چند دور این فرآیند تکرار می‌شود. دو الگوریتم دارد: DES و AES.

ساختار رمزهای فیستل ← به سری وابستگی دارد و چندین دور صورت می‌گیرد.

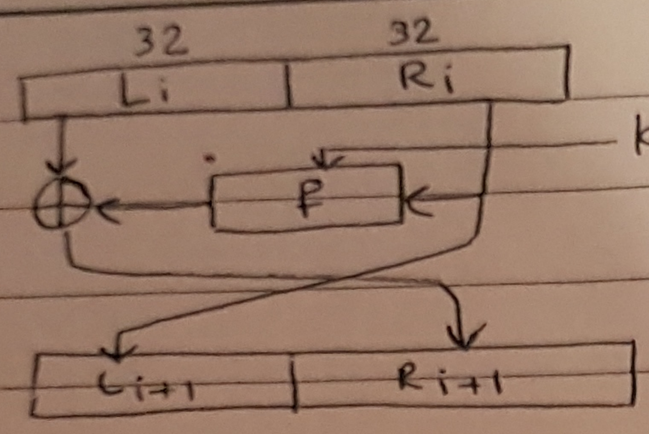
DES ← کلید ۵۶ بیتی دارد. قطعات آن ۶۴ بیتی است و ۱۶ دور طول می‌کشد.

* ۹۶/۷/۱۶

الگوریتم DES: ۱۶ دور عملیات مشابه دارد. الگوریتمی هم برای تولید زیرکلیدهای دورها داریم. ۵۶ بیت دارد می‌شود و ۴۸ بیت کلید آن دور را می‌سازد. (از ۵۶ بیت انتخاب می‌شود).

OMID

Subject:
Year. Month. Date.




$\Rightarrow$ $L_{i+1} = R_i$

$R_{i+1} = L_i \oplus F(R_i, K_i)$

در تابع f ← ابتدا $R_i$ گسترش داده می‌شود و با $K_i$، XOR می‌شود. نتیجه‌اش از 48 بیت به 32 بیت نگاشت می‌شود. درنهایت جایگشتی روی آن انجام می‌شود و از تابع f خارج می‌شود.

برای تولید زیرکلید ← در نیمه‌ی کلید شیفت می‌خورند و درنهایت با جایگشتی تولید می‌شود.

برای تبدیل از 48 به 32 بیت ← یه سری تابع نگاشت داریم که ۶ بیت را به ۴ بیت تبدیل می‌کند. از ۶ بیت ← ۲ بیت سطر و ۴ بیت ستون است. محتوای S-box هم خروجی ۴ بیت را می‌دهد. S-box از قبل طراحی شده است.

حملات DES:

- تحلیل خطی : معادلات را تشکیل بدهیم و با یه سری معادلات خطی تقریبی بزنیم.

اشکال : ① تعداد بالای نمونه‌های اصل رمز شده ② دورهای محدودی را می‌تواند پاسخگو باشد. برای 16 دور brute Force بهتر است. دائماً ضرایب را با احتمالی پیدا می‌کند. در الگوریتم باید احتمالات $1/2 + 1/2$ باشد تا نتوان حدس زد و نمونه‌ها روی احتمال بالاتر متمرکز می‌شود.

- تحلیل تفاضلی : از روی بررسی انتقال ورودی به خروجی به تقریب می‌رسد.

مشکلات DES:

- اندازه کلید : ۵۶ بیت است که اندکی امنیت کم است.

- حملات زمانی : میزان توان یا زمان برای رمزنگاری به تعداد رایانه‌های کلید بستگی دارد. این مشکل در پیاده‌سازی است. از روی توان مصرف شده در نمونه‌ها می‌توان تعداد رایانه‌های کلید را حدس زد.

- الگوریتم های رمزنگاری را می توان در مدهای مختلف به کار گرفت.

Subject:
Year. Month. Date.

- رمزهای قطعه ای:

رمزهای قطعه ای معمولاً در کنار الگوریتم های رمزنگاری به کار می روند. گاهی به کار گرفتن رمزهای قطعه ای باعث ضعیف شدن الگوریتم می شود پس باید در مود مناسب به کار گرفته شود.

- مود کاری:

* ECB: متن را به n بلوک تقسیم کنیم. هر کدام را با کلید رمز کنیم و متن های رمز شده را کنار هم بگذاریم و با همان کلید هم رمزگشایی کنیم. از اشکالات این روش این است که یک متن ساده همواره به یک متن رمز شده ی ثابت تبدیل می شود. این برای تحلیل ترافیک به مهاجم کمک می کند.

* CBC: متن را با یک مقدار اولیه XOR کنیم و حاصلش را با کلید رمز کنیم. حاصل را می سازد که با متن بعدی XOR می شود و در رمزگشایی هم بالعکس صورت می گیرد. در این حالت اگر مقدار اولیه در هر بار متفاوت باشد دیگر متن های رمز شده همیشه یکسان نخواهد بود. در این جا با یک تغییر کوچک به ادامه ی مسیر انتقال می یابد. از اشکالات این روش این است که سازی نمی توان عمل کرد و به هم وابسته است. به علاوه اگر در اثر نویز متن رمز شده با خطا به گیرنده برسد، این خطا تا انتها انتشار می یابد.

* CTR: به شمارنده در حال تغییر را رمز می کنیم و نتیجه اش را با متن اصلی XOR می کنیم. در دریافت هم متن رمز شده را با شمارنده رمز شده XOR می کند. این روش خوبی برای تبدیل رمز قطعه ای به رمز جریانی می باشد. از اشکالات این روش این است که دو طرف باید روی شمارنده همگام باشند و مثلاً اگر packet loss زیاد باشد مدام باید دی سینک شود. مزیت این روش این است که دو طرف سخت افزار یکسانی دارند و موازی هم می شود.

* CFB: مقدار اولیه وارد یک شیفت رجیستر می شود. خروجی اش رمز می شود، بیت های انتخاب می شود و با متن اصلی XOR می شود. متن رمز شده در دفعات بعدی وارد شیفت رجیستر می شود. در طرف گیرنده هم به همین طور است. اشکال این روش انتشار خطا است.

* OFB: مثل CFB با این تفاوت که به جای متن رمز شده، خروجی رمز شده از شیفت رجیستر وارد شیفت رجیستر برای مرحله بعدی می شود. دو طرف مانند هم عمل می کنند و سخت افزارهای یکسان دارند.

Subject:
Year. Month. Date.

۹۶/۷/۲۳ *

- احراز اصالت پیام و رمزنگاری کلید عمومی :

رمزنگاری متقارن کاربرد اصلی اش برای نیازمندی محرمانگی است . منظور از integrity این است که محتوای پیام در طول رسیدن به مقصد عوضش نشود و یا اصالت فرستنده ی پیام زیر سوال نرود گاهی ترتیب و زمان رسیدن پیام هم برای ایان در integrity مهم است . (زمان رسیدن جایی مهم است که مثلاً فردی پیامی را که حتی نمی داند ذخیره می کند و بعداً برای گیرنده دوباره ارسال کند و مزیت کسب کند)

آیا با داشتن روشی که اصالت داده و اصالت فرستنده را چک کند و زمان بخشی از داده ی محتوا باشد و ترتیب هم در محتوا باشد ، این دو را هم می تواند بررسی کند .

* در این فصل مقابله با حملات فعال مدنظر است . (هدف هم integrity است .)

← احراز اصالت با رمزنگاری معمولی (متقارن) : فرض این است که فقط A و B کلید را دارند . با این فرض همین که کلیدش ، محتوا را رمزگشایی می کند از اصالت محتوا مطمئن می شود (اصالت فرستنده در این حالت چک نمی شود (replay)) . فرض می کنیم پیام ها به اندازه ی کافی انرژی دارند که گیرنده می تواند متوجه شود که درست رمزگشایی شده است . اگر هم در وسط راه متوجه نشود و تنها بیت هایش را درست بزند باعث خراب شدن بسیاری از داده هنگام رمزگشایی می شود .

← احراز اصالت بدون رمز کردن پیام : مانند امضای پیام یک برچسب احراز اصالت به پیام الصاق شود . (Message Authentication code (MAC))

$$MAC = F(K_{AB}, M)$$

MAC نیازمندی هایی را برآورده می کند ← اصالت محتوا (چون MAC با محتوا هم کار دارد .)
← اصالت مبدأ
← زمان و ترتیب

رمزنگاری عمومی را می توان برای اصالت به کاربرد اما بهتر است در برخی موارد نباشد ؛
در broadcast ، فراوانگی زیاد مهم نیست و صحت مهم است .
احراز اصالت برنامه ها

* روش های احراز اصالت پیام :

- رمزنگاری عمومی — رمزنگاری غیرمتقارن (public-key E.) (در مقصد با کلید عمومی باز شود و پایه کلید خصوصی رمز شده)
- کپسیه کردن پیام ← کپسیه (H) ← ارسال H با پیام ← در مقصد برعکس

OMID

- تابع Hash (درهم‌ساز) : [illegible]

که کل این فکر به کل بستگی دارد [illegible]

فرو بیاید، اما چون $|U| > m$ است [illegible]

و hash یکسان را ایجاد کند اما فضای حالت [illegible]

ویژگی‌های Hash های امن:

۱) قابلیت اعمال هر اندازه داده

۲) خروجی طول ثابت داشته باشد

۳) محاسبه‌ی آن ساده باشد یعنی [illegible]

* ۴) به دست آوردن $x$ از روی $H(x)$ غیرممکن [illegible]

* ۵) برای هر $x$ نتوان $y$ ای یافت که $H(x) = H(y)$

* ۶) فقط نتوان $(x, y)$ ای یافت که $H(x) = H(y)$

(با اثبات [illegible] Collision [illegible]

[illegible])

[illegible] ۹۴/۷/۱۶

[illegible] ۹۴/۷/۱۶

Subject:
Year. Month. Date.

HMAC ← استفاده از MAC به کمک تابع درهم‌ساز SHA-1 ، برای پروتکل‌های اینترنتی. عموماً تابع Hash سرعت بالاتری از توابع رمزنگاری دارند. در واقع در درهم‌سازی این نوع تابع Hash می‌زنیم. اول $K^+$ (کلید گسترش یافته شده) با یک opad ثابت و معین ⊕ می‌شود و یک بلوک کنار با بلوک b بیتی قرار می‌گیرد و با ورودی IV درهم‌سازی می‌شود و خروجی در ادامه کنار ⊕ حاصل از $K^+$ و opad قرار می‌گیرد و با IV وارد درهم‌سازی می‌شود و در نهایت HMAC را می‌سازد. امنیت HMAC به درهم‌سازی بستگی دارد.

CMAC ← بلوک اول با کلید K رمز می‌شود. خروجی با بلوک بعدی XOR می‌شود و ورودی تابع رمزنگاری را می‌سازد و تا انتها. MSB آخرین بلوک به عنوان مقدار Hash انتخاب می‌شود. کندتر از HMAC است ولی در عوض رمزگیر سخت‌افزاری اضافی برای درهم‌سازی نیاز نداریم.

* Authenticated Encryption :

می‌توان هم درهم‌ساز و هم رمزنگاری باشد. روش‌های مختلف داریم:

Hash-then-encrypt :

MAC-then-encrypt :

Encrypt-then-MAC :

Encrypt-and-MAC :

CCM :

* رمزنگاری نامتقارن (با کلید عمومی) :

در هدف داشتن : حل مسأله‌ی توزیع کلید در متقارن + امضای دیجیتال.

کاربرد هاش ← محرمانگی ← با کلید عمومی رمز شود ← با خصوصی باز می‌شود

امضای رقمی برای ادعای هویت ← با کلید خصوصی رمز شود ← با عمومی باز می‌شود

تبادل کلید ←

در نامتقارن دو دسته کلید داریم ← $K_{private}$ ← فقط خودش دارد

$K_{public}$ ← به همه می‌فرستد.

این دو کلید به گونه‌ای هستند که اگر E با خصوصی بسته باشد D با کلید عمومی باز می‌شود و برعکس

Subject:
Year. Month. Date.

هر فردی یک دسته کلید (key ring) حاوی کلیدهای عمومی همه دارد. کلیدهای عمومی و خصوصی مرتبط هستند اما نمی‌توان از روی عمومی به خصوصی رسید. الگوریتم تولید زوج کلید عمومی و خصوصی باید آسان باشد و در زمان معقول ساخت آن امکان‌پذیر باشد.

* ۹۶/۸/۲

مقایسه‌ی رمزنگاری متقارن و نامتقارن:

کلیدهای کمتری در نامتقارن نیاز داریم. امضای دیجیتال در متقارن نداریم. سرعت متقارن بیشتر از نامتقارن است.

← مسأله‌ی توزیع کلید در رمزنگاری متقارن حل شده است ←

کلید رمزنگاری متقارن ← محرمانه
↖ اصالت

کلید عمومی در رمزنگاری نامتقارن کافی است اصالت داشته باشد. (کلید عمومی مهاجم را به جای کلید عمومی اصلی نباید بتوان جا زد. باید)

پس صرفاً این مسأله برای حل ساده‌تر شده است ولی به طور کامل حل نشده است. فقط در شرط به یک شرط کاهش یافته است.

الگوریتم RSA:

الگوریتمی پرکاربرد و قدیمی است. امنیت الگوریتم بر مبنای این است که مسأله‌ی تجزیه NP-complete است. این مسأله هنوز با قدرت محاسباتی فعلی قابل محاسبه در زمان معقول نیست.

N ← پیمانه  e ← عامل رمزگذاری  d ← عامل رمزگشایی  M ← پیام

RSA encrypt → $x \leftrightarrow x^e \bmod N$

RSA decrypt → $x \leftarrow x^d \bmod N$

تولید کلید RSA:

← p و q دو عدد اول انتخابی‌اند.

← $N=p.q$ و $\phi(N)$ محاسبه شود. ($\phi(N)=(p-1)(q-1)$)

← $1<e<\phi(N)$ به گونه‌ای که $gcd(e,\phi(N))=1$ باشد. ← e به دست آید.

← $d=e^{-1} \bmod \phi(N)$ را حساب کن. ← d به دست آید.

Subject:
Year: Month. Date.

$$y = E(x) = x^e \bmod N$$

$$D(y) = y^d \bmod N = (x^e \bmod N)^d \bmod N$$

$$= x^{ed\phi(N)} \bmod N$$

$$= x^1 \bmod N$$

- توان رساندن باعث می‌شود زمان به p و q وابسته شود ← timing attack ← تولید تأخیرات تصادفی. یا زمان ثابت، اعمال امنی
- ویژگی ضربی دارد که $m_1^e m_2^e = (m_1 m_2)^e \pmod N$ و باعث می‌شود مهاجم بتواند پیام‌های جدید بسازد.

الگوریتم Diffie Hellman :

برای تبادل کلید مورد استفاده قرار می‌گیرد. طرفین q و α را ست می‌کنند. q اول است و α هم‌مولدی برای آن است.

* ۹۶، ۸، ۷

در این الگوریتم برای به دست آوردن $x = dlog_{\alpha,q}(y)$ باید محاسبات پیچیده‌ای را دنبال کرد که برای اعداد بزرگ عملاً غیرممکن است. در نهایت کلید مشترک $= \alpha^{x_1 x_2} \bmod q$

شکل این الگوریتم حمله‌ی meet-in-the-middle است. مهاجم میان تبادلات قرار گرفته است با کلیدهای مهاجم تبادلات صورت می‌گیرد. راه حل آن تصدیق اصالت طرفین است.

الگوریتم‌های خم‌های بیضوی:

زیاد استفاده نمی‌شود ولی مقاومتی آن گویا کمتر از RSA است.

* امضای دیجیتال :

MAC معمولاً بین دو طرف است اما در امضا، یک نفر امضا می‌کند و n نفر آن را و اصالتش را بررسی می‌کنند. با کلید خصوصی و متن امضا تولید می‌شود و طرف دیگر با کلید آن را بررسی عمومی

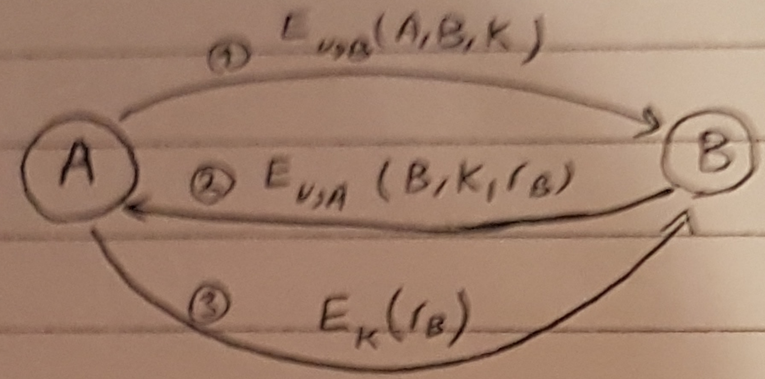
امضا هم در کنار پیام ارسال می‌شود.

MID

Subject:
Year. Month. Date.

اگر از RSA استفاده شود مثلاً کلیدی برهم شده با کلید خصوصی امضا را می‌سازد.

* تبادل کلید: فرض می‌کنیم A و B هر کدام از روش رمزنگاری نامتقارن استفاده می‌کنند و A کلید عمومی B و B هم کلید عمومی A را دارد.




* خطا نتوانستیم که حمله بزنیم! :)

طول عمر توابع درهم‌سازی:

مثلاً SHA-1 ، $2^{64}$ سیکل می‌خواهد. برای حمله باید $2^{60}$ بار تکرار کنیم که در نهایت در کل $2^{74}$ سیکل برای یافتن یک خطا می‌خواهیم. هر هسته $2^{34}$ سیکل در ثانیه دارد و هر سرور 4 هسته دارد. دو سال هم ثانیه داشته باشیم. پس توان محاسباتی هر سرور در یک سال را داریم. باید توان محاسباتی هم را هم 18 ماه دو برابر کنیم و زمان لازم برای شکستن توابع را می‌توان حساب کرد.

* توزیع و مدیریت کلید:

پروتکل Kerberos:

سه هدف داشت: ① Authentication ② accounting ③ Authorization

این پروتکل به جای احراز هویت در هر کارگزار به صورت توزیع شده، یک کارگزار خاص را به احراز هویت اختصاص می‌دهیم.

* 96/8/9

یک Authentication Server اضافه می‌کنیم تا تصدیق اصالت را برای همه‌ی سرورهای موجود دنبال کند. این تصدیق برای این است که هر کاربر یک نام کاربری و پسورد داشته باشد که در همان A.S. ثبت شده است و با همان به صورت امن با همه‌ی سرورهای تحت پوشش تبادل اطلاعات می‌کند. دیگر سرورها نیازی به انجام تصدیق را ندارند.

Subject:
Year. Month. Date.

دیالوگ ساده (۵):

اشکالات این روش:

۱) ID و pass به صورت تمام در طول ابتدا قابل شنود است.

۲) DoS می‌شود زد. (تکرار ارسال ticket هم سرور را بالا می‌برد)

۳) replay هم می‌شود. (به جای کلاینت به سرور)

اگر پسورد را hash ارسال کنیم ← دیگر پسورد خام نباشد ولی باز replay دارد.

یه رندوم هم با پسورد hash کنیم که تاریخ انقضا داشته باشد.

باید بتوان از بلیط استفاده‌ی مجدد کرد ← تا گذرواژه را در یک مدت کوتاه چندین بار وارد نکند.

دیالوگ (۶):

یک سرور TGS برای تولید بلیط ایجاد می‌کنیم. AS فقط یک بار احراز هویت می‌کند.

AS بلیطی برای کاربر برای ارتباط با TGS با کلید کاربر فرض می‌کند که با پسورد او باز می‌شود و بلیط را برای TGS ارسال می‌کند.

در این روش:

۱) هنوز هم امکان replay تا زمان lifetime2 هست. فقط کنترل شده.

۲) سرور جعلی می‌توان داشت.

* ۹۶/۸/۱۴

کریپتوس نسخه ۴ اشکالات عمده‌ی replay حل کرد. همچنین امکان تصدیق هویت دوطرفه را ایجاد کرد.

← مقابله با حمله‌ی تکرار: باید از اینکه خود کاربری فرستد مطمئن شویم. کلید جلسه را برای بسته‌ی تصدیق می‌سازند. در همان بسته‌ای که از AS به کاربری رود یک کلید بین کاربر و TGS ست می‌شود. این کلید یک بار مصرف با تاریخ انقضای محدود است.

با استفاده از این کلید ID و آدرس و زمان را رمز می‌کند. استفاده از این رمز برای تصدیق هویت دوطرفه مفید است.

* DOS روی AS در همان ابتدا ممکن است.

OMID

Subject:
Year. Month. Date.

Realm : ناحیه‌ی مورد حفاظت توسط سرور کربروس . امکان احراز هویت بین قلمروی نیز وجود دارد.
کلید مخفی بین کربروس های مختلف می تواند تبادل شود. باید به تعداد قلمروها کلید تبادل شود
بین آنها تا بتوانند بتوانند به هم مقادیر را ارسال کنند.

هم اکنون کربروس ۵ استفاده می شود که مثلاً وابستگی های به IP و محدودیت زمان را برطرف
کرده است. liftime را با nonce عوض کرده است.

* توزیع کلید عمومی و گواهی های کلید عمومی

یک Master key داریم که برای رمزنگاری session key به کار گرفته می شود. کلید نشست
برای هر ارتباط به کار می رود. secret key هم داریم که زودتر از master key منقضی می شود.
با master ، secret رمز می شود ؛ با secret هم session . کلیدهای مهم تر
کمتر به روز می شود.
کلید عمومی نیاز به محرمانگی ندارد اما باید اصالت داشته باشد. کلید عمومی افراد با یک یک ابزار (سایت)
شناخته شده و مطمئن اصالتش بررسی می شود. گواهی ها یک فرمت استاندارد دارند.

* 96/8/16

توزیع کلید متقارن { محرمانگی / صحت و اصالت کلید }

توزیع کلید عمومی ← اصالت کلید

در کلید عمومی صرفاً لازم است تا اصالت کلید و فرستنده‌ی معتبر آن تأیید شود ← مرکز گواهی کلید عمومی (CA)
certificate مربوط به کلید عمومی اش را امضا می کند. شخص سرویس گیرنده و یا فردی که می خواهد از اصالت
کلید عمومی مطمئن شود، می تواند با این امضا که با کلید عمومی سایت رمزشده چک کند. کلید عمومی CA
را فرض می کنیم روی سیستم هست.
چگونه افراد کلید های عمومی دیگران را به دست می آورند؟ بین CA های مختلف زنجیره‌ی اعتمادی ایجاد شود.
CA ها می توانند سلسله مراتب هم داشته باشند.

هرچند گاهی گواهی ها باطل می شوند که CA به کاربرانش اطلاع می دهد.

- Federated Identity Management : برای این است که کاربران یک هویت در چند سرویس بگیرند
هویت داشته باشند. identity provider ← principle ها و attribute های آنها را نگه
می کند. در سرویس ها علاوه بر هویت، identity افراد هم مهم است.

یه سری استاندارد برای این تشخیص هویت ها وجود دارد:
- OpenID ← مثلاً google، هویت ها را نگه می دارد.

## * Authentication:

- مسأله ی اصلی تصدیق اصالت کاربر است.
کاربر را می توان بر اساس یک یا چند روش تصدیق کرد:
آنچه کاربر می داند ← مثل password
" " دارد ← مثل کارت
" " است ← مثل اثر انگشت
مکان کاربر ← معمولاً مقش نیست و ترکیب با بقیه می شود.
سیستم احراز هویت ← (A, C, F, L, S)

Subject:
Year Month Date

- انواع حمله علیه رمزنگاری

[illegible]

[illegible]

- [illegible]

- Ciphertext only : کلیه متن رمز شده

- Known plain text : کلیه جفت های متن اصلی و متن رمز شده

- Chosen plain text : کلیه متن های اصلی که توسط آنالیزگر انتخاب می شوند و متن رمز شده آن ها استخراج می شود

- Chosen ciphertext : کلیه متن رمز شده که آنالیزگر می تواند رمزگشایی کند

- Chosen text : کلیه متن های اصلی که آنالیزگر می تواند رمز کند

- Brute Force Attack

[illegible]

* ۹۶/۷/۱۱

- امنیت مطلق ← مثل انتخاب [illegible] که هیچ اطلاعاتی [illegible] نمی دهد.
  ← [illegible]

- امنیت محاسباتی ← هزینه محاسباتی شکستن اطلاعات رمز بیشتر از ارزش اطلاعات باشد

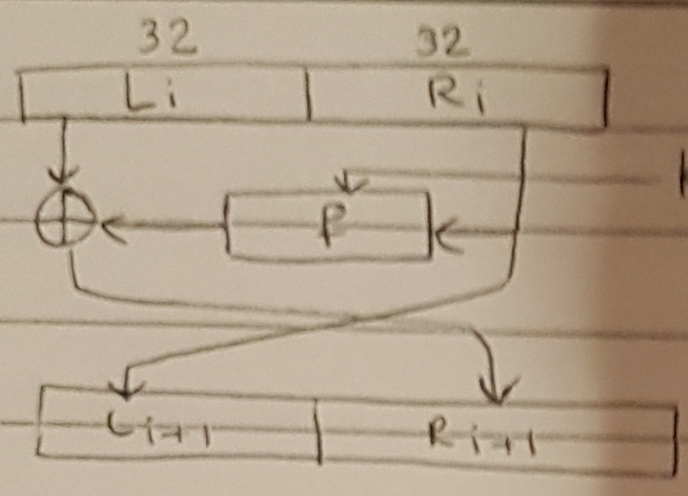
* [illegible] : [illegible] (Kerckhoffs) [illegible]

[illegible]

[illegible]

$P(A|B) = P(A)$

Subject:
Year. Month. Date.




$k_i$ (48bit) $\Rightarrow$ $L_{i+1} = R_i$

$R_{i+1} = L_i \oplus F(R_i, k_i)$

در تابع F ← ابتدا Ri گسترش داده می‌شود و با Ki، XOR می‌شود. نتیجه‌اش از 48 بیت به 32 بیت نگاشت می‌شود. در نهایت جایگشتی روی آن انجام می‌شود و از تابع F خارج می‌شود.

برای تولید زیرکلید ← در نیمی کلید شیفت می‌خورند و در نهایت با جایگشتی تولید می‌شود.

برای تبدیل از 48 به 32 بیت ← یه سری تابع نگاشت داریم که 6 بیت را به 4 بیت تبدیل می‌کند. از 6 بیت ← 2 بیت سطر و 4 بیت ستون است. محتوای S-box هم خروجی 4 بیت را می‌دهد. S-box ها از قبل طراحی شده است.

DES تحلیل:

- تحلیل خطی: معادلات را تشکیل بدهیم و با یه سری معادلات خطی تقریبی بزنیم.

اشکال: (1) تعداد بالای نمونه‌های اصلی ارزشمند (2) دو رهاله ورودی را می‌تواند پاسخگو باشد. برای 16 دور brute force بهتر است. مثلاً ضرایب را با احتمالی پیدا می‌کند. در الگوریتم بایاس احتمالات $\frac{1}{2}+\epsilon$ باشد. با افزایش تعداد دور، دو گزینه مهاجم روی احتمال بالاتر متمرکز می‌شود.

- تحلیل تفاضلی: از روی بررسی انتقال ورودی به خروجی به تقریب می‌رسد.

مشکلات DES:

- اندازه کلید: 56 بیت است که اندکی امنیت کم است.

- حالت رمانی: میزان توان یا زمان برای رمزنگاری به تعداد بایت‌های کلید بستگی ندارد. این مشکل در پیاده‌سازی است. زمان توان مصرف شده در رمزنگاری و توان تعداد بایت‌های کلید یا

Subject
Year Month Day

Subject:
Year. Month. Date.

Key → الگوریتم رمزنگاری از یک داده‌ی کوچک برای رمزنگاری استفاده می‌شود

encipher → تبدیل متن عادی به متن رمزشده

decipher → بازیابی داده‌های متن عادی از متن رمزشده

cryptography → مطالعه‌ی اصول و روش‌های رمزنگاری

cryptanalysis → مطالعه‌ی روش‌ها و اصولی برای بازیابی متن اصلی بدون داشتن کلید

cryptology → cryptography + cryptanalysis

متقارن ← چون دو طرف گیرنده و فرستنده دارای یک کلید یکسان هستند تابع رمزنگاری هم معکوس پذیر است به گونه‌ای که حتماً نیاز به کلید باشد تا تابع وارون به درست آید

نیازمندی‌ها ← الگوریتم رمزنگاری قوی

کلید سری که تنها فرستنده و گیرنده از آن آگاه باشند

در رمزنگاری رمز برای این است که الگوریتم را همه می‌دانند تا طرف مقابل هم بتواند معکوس کند بنابراین برای توزیع کلید نیاز به کانال امن داریم

مثال: A می‌خواهد روی صندوقچه نامه‌ای بفرستد به B بفرستد. هر کدام یک قفل خودش و کلیدش را دارد. باید سه بار برود و برگردد: A قفل کند، B قفل کند، A باز کند، B قفل خودش را باز کند. البته man in the middle امکان است: پستچی با دریافت از A، قفل خودش را به صندوق B می‌زند و به A برگرداند و او باز می‌کند. حال نقل قفل پستچی روی صندوق B را در دسترس دارد. می‌تواند برای اینکه شکرگزار شوند برای B هم نقش A را بازی کند.

برای تقسیم‌بندی الگوریتم‌های رمزنگاری می‌توان براساس عملیات انجام شده، تعداد کلیدهای مورد نیاز، روش پردازش متن:

- عملیات انجام شده ← جایگزینی / ضرب / جمع
- تعداد کلیدهای مورد نیاز ← یک کلید عمومی / دو کلید عمومی یا خصوصی
- پردازش متن پیام ← بلوکی ← داده‌ها جمع می‌شود و بلوک بلوک رمز می‌شود / جریان داده‌ای ← با جریانی رندم بیت بیت OMISS

جریان رندم با یک کلید مشترک است.

Subject:
Year. Month. Date.

- سرویس امنیتی:

هدف مقابله با حملات امنیتی است

تعریف: RFC2828 یک سرویس ارتباطی با پردازشی برای حفاظت از منابع سیستم

پروتکل استاندارد X.800 ۷ سرویس داریم

- تصدیق اصالت: اطمینان از تصدیق ادعا شده مورد ارسال کننده

- کنترل دسترسی: جلوگیری از دسترسی غیرمجاز

- محرمانگی داده: جلوگیری از افشای اطلاعات

- صحت داده: ارسال و دریافت بدون تغییر صورت گیرد

- عدم انکار: ارسال کننده و دریافت کننده نتواند ارسال یا دریافت را انکار کند

- دسترس‌پذیری: در دسترس بودن منابع

برای هر کدام از این سرویس‌ها مکانیزم‌های متفاوتی داریم اما معمولاً رمزنگاری جزء مکانیزم‌های بسیار رایج است.

در امنیت یک دیدگاه امنیتی برای شبکه ارائه شده است که طی آن یک پیام استراتژیکی مشروع بین دو طرف کانال شبکه تبادل می‌شود. و یک سیستم اطمینان برای بررسی پیام‌های امنیتی بر یک Gatekeeper میزبان دارد. معمولاً چک کردن بودن پیام‌ها در کانال دارد.

۹۶/۷/۲۴ *

- رمزنگاری متقارن:

یکی از روش‌های محافظت از داده‌های ما یعنی data می‌باشد. رمزنگاری یعنی سیستم فرآیند پروتکل با الگوریتم رمزنگاری چیست.

plain text → پیام اصلی و سالم (خط‌کشی تغییر)

cipher text → پیامی که تغییرات کرده است و خط‌کشی است

cipher → الگوریتمی که برای رمزکردن ارسال به رمزشده استفاده می‌شود