**** اخطار! ****

این جزوه تاییدیه استاد را ندارد و جزوه ای شخصی است.

این جزوه میتواند ناقص و یا اشتباه باشد.

تهیه این جزوه یک کار داوطلبانه بوده است

و حضور نداشتن شما در کلاس، مسئولیتی را برای بنده ایجاد نمیکند

مفاهیم رمز نگاری

پیش نیاز:

۱- ساختمان داده رمز نگاری چیست و چه ساختار داده ای دارد

۲- تئوری احتمالات

۳- تئوری پیچیدگی زمان های الگوریتم ها

۴- ریاضیات عددی و تئوری اعداد (نیوتون)

۵- جبر رابطه ای

متن اصلی درس

۱- رمز نگاری متقارن: شامل طراحی الگوریتم های DES, 3DES, AES

طراحی الگوریتم های کلید مشترک

۲- توابع درهمساز و ساختارهای داده ای آنها Hashing Table

توابع MD5, SHA1

تحلیل تطبیقی توابع

۳- سیستم های رمزنگاری کلید عمومی
الگوریتم های دفی هلمن و RSA
معادلات هذلولی روی رمز
توابع الجمال
خم های بیضوی
امضای دیجیتال

۴- مدیریت کلید و ساختار عمومی کلید عمومی و گواهینامه های مربوط به کلید عمومی و کاربرد آن ها در نرم افزار

۵- تحلیل ریاضی رمز شامل تحلیل رمز تفاضلی، تحلیل رمز خطی و تحلیل رمز تفاضلی کوتاه شده

۶- قراردادها و استانداردهای امنیتی و کاربرد استانداردهای رمز نگاری در VPN ها، پروتکل های SSL و IPSEC ، مجوزهای مدیریت و زیرساخت آنها PMI و کاربرد این استاندارد ها در دسترسی Access Controll تجارت الکترونیک و کارت های هوشمند

منابع

Cryptography and network Security – Printers Hall by Stalling

Applied Cryptography – John Willy – Schneire بحث ریاضی رمزنگاری

Hand book of Applied Cryptography – menezes

Stinson, Cryptography: theory and Practice P.H

Cryptography

رمزنگاری علمی است جهت تغییر حالت طبق یک ساختمان داده روی داده های موجود که این داده ها بنابر مصلحت خاصی باید تغییر پیدا کند و به کمک همان ساختمان داده از حالت رمز بازگردانده شود.

این عملیات را Encryption و Decryption می گویند.

چرا از رمز نگاری استفاده میکنیم؟

علم رمزنگاری باعث میشود تا داده مبادله شده بین دو طرف نزد طرف دیگر قابل شناسایی و دستیابی نباشد و یا حتی الامکان مخفی باشد که در حین مبادله دچار تغییر و یا دزدیده شدن و یا دسترسی نابه جا قرار نگیرد.

در علم رمزنگاری ورودی یک متن ساده است و خروجی متن رمز شده. اگر از الگوریتم رمز طبق یک اصل ریاضی و با ساختار داده ای درست استفاده شود متن خورجی رمز شده را میتوانیم از رمز در آوریم و به متن معمولی باز گرداندیم.

مرحله تبدیل plaintext به cipher text را Encryption و مرحله بازگشت را Decryption گویند.

تذکر: در حالت اول Encryption، دیتا یا پیام به Stored Data تبدیل میشود.
و در حالت دوم Stored Data به Original Data تبدیل میشود.

هر رمزنگاری به یک کلید نیاز دارد. این کلید میتواند بین دو طرف مشهود باشد و یا اینکه میتواند به صورت اختصاصی برایآنها تعریف شود.
اگر مشهود نباشد به آن کلید خصوصی Private Key گویند و برای استفاده از آن یک کلید دیگر نیاز داریم. چون کلید خصوصی برای هر شخص نزد خودش باقی می ماند.
اما اگر کلید عمومی باشد احتمال لو رفتن آن نیز وجود دارد.

تذکر: به افرادی که در رمزنگاری کار میکنند Cryptologist میگوییم. این افراد در حوزه رمزنگاری و باز کردن رمز و تعیین ساختار های رمز از تعاریف ریاضی استفاده میکنند و به کمک آنها میتوانیم برای امنیت داده های خود در مسیر و یا پنهان سازی داده ها و مجوزهای دستیابی به آنها استفاده نماییم.

نکته: انواع Cipher

سایفر ها بر اساس نوع کلید تعریف می شوند. اگر کلید خصوصی باشد سایفر از نوع خصوصی Secret میباشد.
اگر کلید عمومی باشد سایفر به عنوان کلید غیراشتراکی استفاده میشود.

هنگامی که دو طرف مبادلات پیام را انجام میدهند معمولا این مبادلات بر سر پرسش و پاسخ هاست و یک تصمیم گیری را پشتیبانی میکند و ممکن است در اثر این مبادلات، متن، عدد، تصویر، نرم افزار و ... رد و بدل شود.
پس نتیجه میگیریم که پیام مبادله شده هر نوع دیتا یی میتواند باشد.

در قالب های رمزگذاری یک الگوریتم، عملیات رمز را انجام میدهد و الگوریتم دیگر عملیات باز کردن را.

Encryption Algorithm E

Encryption Algorithm D

این الگوریتم برای اجرای E و D از کلیدی به نام K استفاده میکند.
اجرای عملیات با تعریف زیر صورت میپذیرد.

* Encryption a plaintext p using a key k & an Encryption Algorithm E C=E(K,P)

* Decryption a Cipher text C using the Same key K and the matching Decryption Algorithm D P=D(K,C)

* Note: P = D(K,C) = D(K,E (K,P))

رمز گذاری سزار به عنوان رمز گذاری نمونه:
این رمزگذاری یکی از انواع ساده ترین انواع تبدیل دیتا روی متن های ساده یا رشته های متنی و یا عددی است و یک کلید مشترک برای عملیات رمزگذاری وجود دارد. در ساده ترین حالت این رمزگذاری Text معمولی به اندازه ۳ حرف به عقب و یا چپ شیفت میشود. در این رمزنگاری متن رمز شده برای بازشدن رمزش از الگوریتم مرتبط استفاده میکند و به اندازه ۳ حرف به راست شیفت پیدا میکند.

در نظر داشته باشید که ساختمان داده توابع D و E یکی است.

در شکل سزار، اگر کلید شیفت ۳تایی باشد و حروف اصلی ۲۶ حرف انگلیسی باشد، Cipher text سه حرف به عقب شیفت پیدا میکند که میتوانید جمله Treaty Impossible را به شکل زیر ببینید:

K=3	T	R	E	A	T	Y	I	M	P	O	S	S	I	B	L	E
	W	U	H	D	W	B	L	P	S	R	V	V	L	E	O	H

از آنجائیکه رمز فوق یک رمز کلاسیک است. با استفاده از حملات آماری مثل دانستن فراوانی نسبی تجمعی و فراوانی تجمعی میتوانیم این رمزها را بشکنیم و این نوع رمزها با دانستن تعداد تکرار حروف و دانش آماری روی حروف به نسبت های مختلفی شکسته میشود. در متن زیر درصد استفاده از حروف و تعداد تکرار آنها یا فراوانی تجمعی آنها نوشته شده است.

A	٪۷,۴۹
B	٪۱,۲۹
C	٪۳,۵۴
D	٪۳,۶۲
E	٪۱۴,۰۰

** جلسه سوم **

در رمز گذاری سزار الگوریتم Encode به شکل زیر میباشد:

Visual basic Encoding:

```
For i = 1 To Len(plain)
 newchar = Asc(UCase$(Mid$(plain, i, 1))) + key
 if newchar > 90 Then newchar = newchar – 26
 Cipher = Cipher & Chr$(newchar)
Next

MsgBox Cipher
```

برنامه Decoding نیز مانند برنامه Encoding است با این تفاوت که مرحله Shift را در جهت عکس انجام میدهد:

```
For i = 1 To Len(plain)
  newchar = Asc(UCase$(Mid$(Cipher, i, 1))) – Key
  if newchar < 65 Then newchar = newchar + 26
  decode = decode & Chr$(newchar)
Next

MsgBox decode
```

در این رمزگذاری طبق برنامه فوق، حروف کوچک و بزرگ در نظر گرفته شده اند، و اگر هنگام تبدیل یا جایگذاری چرخشی انجام شود بر اساس متغییر Key خواهد بود و کنترل تعداد حروف طبق متغییر Key با بیست و شش حرف انگلیسی در نظر گرفته میشود که اگر از بیست و شش حرف بالاتر شد، در الگوریتم در نظر گرفته میشود.
جایگذاری میتواند، سه حرف، پنج حرف و ... و یا کلمه باشد. در اصل الگوریتم سزار عبارت الفبای انگلیسی به صورت زیر جایگذاری میشود

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

در روش الگوریتم بیان شده، **!26** حالت رمز داریم که چون بزرگی و کوچکی حروف را در بر میگیرد، کار شکست رمز را سخت میکند.

تمرین: الگوریتم سزار را با فرض زیر برنامه نویسی کنید که عبارت: HELLO ALANIA را به رمز تبدیل کند، با فرض اینکه به جای هر حرف کلمه MAX قرار گیرد و K=3 باشد. (بدین صورت که اگر A جایگزینش MAX شود، جایگزین حرف B با کلید سه تایی کلمه ای خواهد بود که هر حرفش سه حرف بعد از هر حرف MAX خواهد بود - استفاده از آرایه دو بعدی)

در رمزنگاری جانشین، کلید رمز حداقل دو حرفی باید باشد. اگر بخواهیم از رمزگذاری جایگشتی یا Transpost استفاده کنیم در آن صورت، باید دید ماتریسی روی plaintext وجود داشته باشد.

سزار را اگر با این نحوه رمزگذاری رمز کنیم، کشف رمز طولانی تر شده و جایگشت های آن روی ستون های جدول تعریف خواهد شد.
تمرین قبل از این نوع رمزگذاری است. این رمزگذاری از ماتریس transpose استفاده میکند.

روش دیگر رمزگذاری سزار، با کمک بهم ریزی محتوا one Time Pad:

در روش بهم ریختگی پیام، معمولا یک رشته حرفی، تصادفی انتخاب میشود که طولش با طول متن یکی است، سپس کد ASCI حروف متن اصلی با کد ASCI حروف کلید رمز XOR میشود. معمولا اگر کلید از عبارت ها یا حروف پراکنده تری استفاده کرده باشد، در آن صورت تکرار حرف نخواهیم داشت. اگر موقع برگرداندن رمز، از خود XOR استفاده کنیم، به راحتی رمز شکسته میشود.

در سیستم های فوق، معایبی وجود دارد:

۱- ممکن است طول کلید آنقدر زیاد شود که نتوانیم آنرا به خاطر بسپاریم.
۲- حجم اطلاعات شامل، طول متن + طول کلید میشود.
۳- کلید باید در یک کانال امن قرار گیرد چون اگر لو برود، رمز هم لو رفته است.

برنامه زیر نشان دهنده رمز نگاری به روش One Time Pad میباشد:

```
PRIVATE SUB FORM_LOAD()

  DIM PLAIN$,KEY$,CIPHER$, I%, DECODE$

  PLAIN = "HELLO, HOW ARE YOU"
  KEY = "AJUWERXSDFJHSKDF1KS"

//* ENCODE SECTION

FOR I = 1 TO LEN(PLAIN)
  CIPHER = CIPHER & CHR$(ASC(MID$(PLAIN, I , 1)) XOR ASC(MID$(KEY, I ,1)))

NEXT I

MSGBOX CIPHER

//* DECODE SECTION

FOR I = 1 TO LEN(PLAIN)
  DECODE = DECODE & CHR$(ASC(MID$(CIPHER, I, 1)) XOR ASC(MID$(KEY, I, 1)))
NEXT I

MSGBOX DECODE

END SUB
```

در برنامه فوق تابع MID از اولین اندیس رشته حرفی شروع میکند، که رشته حرفی میتواند Plain یا CIPHER باشد و تا آخرین حرف رشته حرفی، یکی یکی جلو میرود. این کار را به کمک I درون حلقه FOR و اسکی کد آن را با کمک تابع ASC بیرون میکشد و با اسکی کد کلید که به همین روش به دست آمده است XOR میکند. این رمزگذاری دو طرفه خواهد بود.

رمزنگاری دو اصل مهم دارد: افزونگی و تازگی

در افزونگی در پیام های رمز شده تکرار وجود نداشته باشد. که هکر یا کرکر متوجه شکست رمز و راحتی آن نشود. افزونگی ممکن است در کد بندی رمز پیدا شود. ممکن است در درهم سازی پیدا شود.

در تازگی پیام همیشه باید عبارات جدید وجود داشته باشد و نحوه ارسال پیام روی عبارات قدیمی ثابت نشده باشد.

جلسه چهارم

الگوریتم Hash:

Hash ساختمان داده ای است که روی آرایه ها کار میکند و میتواند، با جا به جایی عناصر آرایه ها و تولید رشته های بیتی از روی آنها در پنهان سازی دیتا استفاده شود. به Hash ، ساختمان داده در هم نیز گویند و الگوریتم های آن معمولا از نوع رمزگذاری های یک طرفه است. در این الگوریتم ها میتوانیم به تعداد بیت زیاد، دیتا برای کلید داشته باشیم و معمولا پردازش آنها سنگین است. نمونه رمزهایی که از Hash استفاده میکنند رمزهای MD5، SHA1، SHA2، Tiger، SHA3 ، GOST3411 میباشد.

در رمزگذاری SHA1 معمولا پیام ورودی با هر طول دلخواهی تا سقف 2^{64} بیت وارد ماشین رمز میشود و از آن یک رشته ۱۶۰ بیتی خارج میشود. که به آن Message Digest گویند. خاصیت این الگوریتم در آن است که اگر کسی یک پسورد را وارد کند شکل کدگذاری شده آن ثابت و منحصر به فرد خواهد بود.

الگوریتم های Hash عموما برای تایید هویت و شناسایی فایل، تایید پیام دریافتی، نگهداری رمز عبور به صورت کد شده و غیره استفاده میشود.

SHA1 و MD5 به هم نزدیک هستند و در آن ضعف نسخه های قبلی تر پوشیده شده است. SHA2 با SHA1 کلهم فرق دارد.

مدت زمان شکسته شدن رمز SHA1 بستگی به طول پیام و پردازش ماشین رمز دارد که معمولا چند ده یا صد سال طول میکشد.

در الگوریتم SHA1 چهار فرایند وجود دارد:

۱- افزودن بیت های لایه گذاری
۲- افزدون طول پیام
۳- ایجاد و فراخوانی توابع پردازشی
۴- مقداردهی اولیه بافر برای پردازش، ورودی و خروجی و ذخیره سازی

در این رمز گذاری، اصولی به نام Padding وجود دارد که به کار گیری بیت های نرم کننده است. این روش روی ۴۴۸ بیت پیام که کمتر یا بیشتر باشد تاثیرگذار است و باید مضربی از ۵۱۲ بیت باشد. سپس در ادامه نرم کردن بیت ها به آن یک مقدار ۶۴ بیتی اضافه میشود که این پیام را قابل افزایش طول میکند. در عملیات بافر پنج متغییر در نظر گرفته میشود به نام های A,B,C,D,E که این پنج متغییر هر کدام ۳۲ بیت از ۱۶۰ بیت پیام را خواهند داشت. در این صورت ماشین رمز با کمک تابع های فشرده ساز متغییرها را داخل بلاک های ۵۱۲ بیتی قرار میدهد و خروجی آن توابع F1 تا F4 میشود که شامل چهار دور پردازشی هستند که این توابع را توابع پردازشی نیز گویند.

در خلال اجرای توابع پردازشی خروجی کلماتی خواهد بود، از w_0 تا w_{79} که رمز شده پیام خواهد بود.

** جلسه پنجم **

رمزگذاری AES

این رمزنگاری به صورت رمزنگاری پیشرفته مطرح میشود، پیشینه آن از موسسه NIST می باشد، در سال ۲۰۰۱، از روی رمزگذاری AES ابتدای رمزگذاری جدیدی برای استانداردسازی اطلاعات موجود استفاده شده است، این رمزگذاری به موارد زیر نیاز دارد:

۱- امنیتی معادل رمزهای Triple DES
۲- امضای دیجیتال با کمک 3DES و انعطاف پذیر تر از آن
۳- تعریف یک بلاک رمزگذاری شده Symmetric یا متقارن
۴- طول بلاک ۱۲۸ بیت باشد
۵- کلید طول ۱۲۸ یا ۱۹۲ یا ۲۵۶ بیت داشته باشد

این رمزگذاری دارای محدوده عملیاتی زیر است:

الف) امنیت بالا
ب) پیچیدگی در انعطاف
ج) حافظه بالا
د) سخت افزار قوی

در رمزگذاری AES ساختار الگوریتم یک بلاک ۱۲۸ بیتی را (یا ۱۶ بایتی) را به عنوان ورودی میگیرد، و آن را در یک آرایه ای به نام State Array قرار میدهد.

کلید وارد شده در یک بلاک به دو بخش یا دو کلمه، چهار بایتی، تقسیم میشود Schedule Keyword
این کلید های 128 بیتی به کلمه های ۴۴ تایی تقسیم میشوند و در داخل یک ماتریس مربع که وضعیت و حالات را توصیف میکند، قرار میگیرد. در مرحله جدید، عملیات دوران و تغییر حالت انجام میشود، بدین صورت که اجرای پردازش های رمزگذاری توسط یک تابع به نام تابع NR انجام میشود. ممکن است چندین بار بر اساس سایز کلید این تابع فراخوانی شود، سپس این تابع در هر فراخوانی، چهار وضعیت را تولید میکند

۱- وضعیت پیدا کردن بایت های جزئی sub byte
۲- حرکت درون سطرها shift row
۳- ترکیب ستون ها Mixed column
۴- اضافه کردن کلید که از تابع NR به دست می آید add round key
۵- در مرحله بعد کلمه رمز شده با همین کلید شروع میشود، سپس همه بخش های کلمه رمز شده به حالت قبلی خود تغییر حالت میدهد و تابع Mixed تغییر حالت ها را به وضعیت نهایی تبدیل میکند. معمولا برای کلید ۱۲۸ بیتی، ۱۰ عمل Round انجام میشود.

عملیات sub byte شامل بخش های زیر است:

۱- ایجاد جدول پردازشی S Box که بستگی به طول کلید دارد
۲- هر بایت از وضعیت با بایت دیگری جایگزین میشود و در جدول می نشیند.
۳- جدول با دو تغییر حالت تعریف شده، محتویاتش ثابت میشود
۴- یک ماتریس هشت در هشت، ایجاد میشود که به کمک بایت های موجود در مرحله دو، یعنی State Byte اندیس گذاری خواهد شد.

در تابع Shift Row عملیات زیر انجام میشود:

۱- ایجاد وضعیت برای تغییر حالت به کمک سطر ها در جدول حالت بر اساس طول کلید
۲- دو سطر اول، یا یک سطر اول را شیفت ندهید

۳- سطر دوم، یک شیفت چرخش به چپ یک بایتی دارد
۴- سومین سطر به صورت چرخشی به چپ دو بایت شیفت پیدا میکند
۵- چهارمین سطر به صورت چرخش به چپ سه بایت شیفت پیدا میکند

در تابع Mixed Column موارد زیر را داریم

۱- در این تابع، ستون های جدول حالت، تغییر حالت پیدا میکند
۲- پردازش میتواند به صورت یک چند جمله ای، با کمک آرایه حالت ها، توصیف شود و در ماتریس مربوطه قرار میگیرد.
۳- پردازش این بخش، به کلید وابسته نیست

در تابع Add Round Key

۱- این تابع، تغییر حالت را با کلید های به دست آمده از Sub Byte، تولید میکند، سپس با عملیات XOR بیتی ستون های آرایه حالت را با کمک کلید پردازش می کند.

این چهار بخش بیان شده، که چهار تابع مهم می باشند، عملیات رمزگذاری AES را انجام میدهند. برای رمز برداری از حالت معکوسی استفاده میکنیم، تا کشف رمز موجود انجام شود.

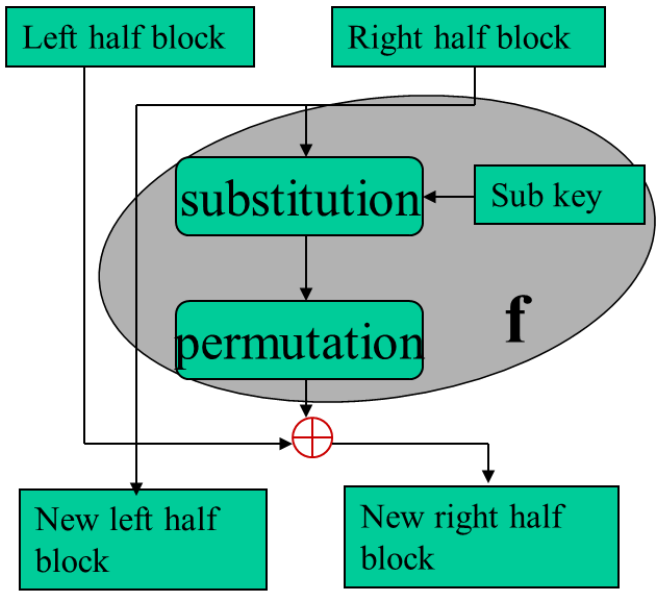
** جلسه ششم **

رمزگذاری DES و 3DES

این رمزگذاری پایه رمزگذاری AES است، در رمزگذاری DES از یک کلید ۶۴ بیتی استفاده میشود، ولی به صورت کلی ۵۶ بیت از آن مورد استفاده قرار میگیرد، چند بیت برای Parity انجام میشود، در 3DES دو ۶۴ بیت دیگر نیز استفاده میشود و از هر کدام از آنها بیت برای اولویت در نظر گرفته خواهد شد. رمزگذاری یا رمزبرداری این روش، در حالت Code Boot روی بلاک های ۶۴ بیتی اجرا میشود. در رمزگذاری DES معمولا Cypher از نوع متقارن است و از تابع Round روی هر بلاک میتوان، پیچیدگی الگوریتم را اجرا کرد.

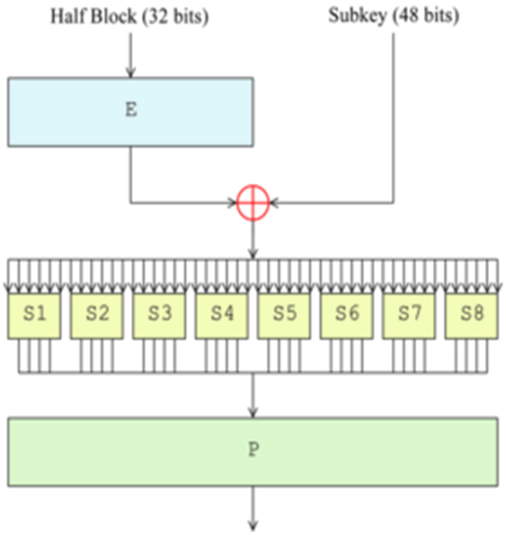
DES اولیه توسط IBM در ۱۹۷۶ ایجاد شد، و در ۱۹۹۰ به ترتیب بهینه ای عمل نمود، این رمزگذاری منحصر به NSA و برای رمزکردن متون و مستندات آن می باشد. عملکرد Round بدین صورت است که از ۶۴ بیت، ۴۸ بیت به عنوان SubKey استفاده میشود و به دو قسمت تقسیم میشود که به آنها بلاک های نیمه راست و نیمه چپ گویند. پس از پیدا کردن این دو بلاک، بلاک نیمه چپ با کمک تابع substitution و افزودن subkey تبدیل به یک بلاک جدید می شود، این بلاک توسط تابع Permutation تبدیل به بلاکی میشود که با نیمه راست XOR خواهد شد. سپس جای بلاک های نیمه راست و چپ تغییر پیدا میکند. خروجی این ماشین تابع F است که مجموعه ۶۴ بیت جدید را در یک Encoder نشان خواهد داد.

در تفاوت DES و 3DES در Subkey آنها است. در DES دو کلید K1 و K2 داریم، در 3DES در مرحله اول کلید K1 و مرحله دوم کلید K2 و در مرحله سوم کلید K1 استفاده میشود، سایز 3DES بر حسب سایز کلید ۱۱۲ بیت خواهد شد. یکی از مهمترین مشکلات رمزگذاری DES کوتاه بودن کلید آن است.



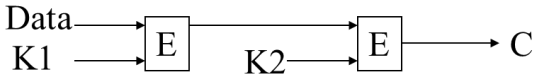
A Round of Encoding a block in DES (repeat 16 times)

هنگام پیاده سازی خروجی XOR ماشین های S1 تا S8 خروجی بلاک P را میدهند که این بلاک نهایتا Function F را مشخص میکند.



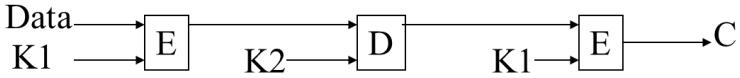
در این الگوریتم، میتوانیم از کلید های K1, K2 که هر کدام هشت کارکتر هستند استفاده کنیم، Plain Text میتواند هشت کارکتر باشد، Cypher Text هشت کارکتر و همچنین در برنامه میتوان یک Recover Text گذاشت که آن نیز هشت کارکتری است. با توجه به اینکه 3DES سه کلید استفاده میکند و DES دو کلید. به شکل زیر:

- Two-Key DES



  – Total key size is 56x2=112 bits; but the effective key size is only 57 bits only!

- Triple DES (3DES)



  – This is a secure solution with effective key size of 112 bits

برای شکستن کلید های کوتاه دو به توان ۵۵ حالت میانگین سعی و خطا وجود دارد، برای CPU های کم سرعت مجبور هستیم که رمزهای کوتاه بگذاریم و شکستن رمز آنها سریع انجام میشود.

3DES میتواند جداول AES را نیز شکل دهد. در 3DES برای پیاده سازی برنامه به زبان C میتوانیم تعاریف داده ای زیر داشته باشیم.

Triple-DES Example

```
unsigned char key1 [8];
unsigned char key2 [8];
unsigned char key3 [8];
unsigned char plaintext [8];
unsigned char ciphertext [8];
unsigned char recoverd [8];
tripledes_ctx context;
```

در این الگوریتم یک ساختار تعریف میشود که این ساختار میتواند به تولید رمز در تابع F کمک کند

ساختار DES

```
*/
  Typedef struct_des_ctx
{
   int mode;
   u32 encrypt subkeys [32];
   u32 decrypt subkeys [32];
}
des_ctx [1]
/*
```

ساختار 3DES

```
/*
*  Encryption/Decryption Context of Triple_DES
*/
     Typedef Struct_Tripledes_ctx
     {
      int mode;
      u32 encrypt_subkeys [96];
      u32 decrypt_subkeys [96];
```

با توجه به این دو ساختار میتوانیم، Sbox را به صورت یک آرایه ۳۲ بیتی، ۶۴ عنصری تعریف کنیم.

```
Static u32 sbox1[64]
```

Sbox 2 به همان ترتیب sbox1 تا sbox8 به همان صورت تعریف می شود:

متغییر leftkey و rightkey نیز به صورت آرایه های ۱۶ عنصری تعریف میشود.

```
U32 leftkey_swap[16]
u32 rightkey_swap[16]
```

تابع round نیز از یک جدول استفاده میکند، برای Encrypt و یک جدول برای Decrypte، این جداول ۱۶ عنصری هستند و از جنس بایت

```
Static byte encrypt_rotate_tab [16]
Static byte decrypte_rotate_tab[16]
```

تابع Round در DES به صورت زیر عمل میکند

```
Define DES_ROUND (from, to, work, subkey)
work = ((from<<1) | (from>>31)) ^ *subkey++;
to ^=sbox8 [  work & 0x3f ] ;
to ^=sbox6 [ (work >>8) & 0x3f ] ;
to ^=sbox4 [ (work >>16) & 0x3f];
to ^=sbox2 [ (work >>24) & 0x3f];
work = ((from>>3) | (from<<29) ^ *subkey++;
to ^=sbox7 [  work & 0x3f ];
to ^=sbox5 [ (work>>8) & 0x3f ];
```

```
to ^=sbox3 [ (work>>16) & 0x3f ];
to ^=sbox1 [ (work>>24) & 0x3f ];
```

در این برنامه تابع DES key Schedule دارای پارامترهای ورودی زیر می باشد:

```
DES_Key_Schedule (const byte * rawkey, u32 * subkey, int mode)
```

جلسه هفتم تقسیمات ریاضی ماژولار موجود نیست

**** جلسه هشتم ****

RSA : یکی از رمزنگاری های کلید عمومی است، در این رمزنگاری نیز مانند رمزنگاری های قبل از تقسیمات و ضرب متوالی استفاده میشود، در اثر این تقسیمات یک جدول تعریف میشود که در آن جدول مقادیر مربوط به توابع Return و Extend قرار خواهند گرفت. این الگوریتم بر مبنای الگوریتم اقلیدسی یا هندسه اقلیدسی است، که از تابع بزرگترین مخرج مشترک استفاده میکند. در RSA رمزنگاری با کلید مشترک انجام میشود و به عنوان رمزنگاری متقارن نیز معرفی میگردد، در این رمزنگاری میتوان از یک کلید نیز استفاده کرد، شروع رمزنگاری ۱۹۷۷ و مقدار عدد Integer ای که آن قبول میکند، اعداد صحیح بزرگ است. در این رمزنگاری معمولا عدد صحیح بزرگ تقسیم توانی بر یک عدد اول را نشان میدهد. O ((log n) ^3)

مرتبه اجرایی این الگوریتم، با عمل فاکتوریل ـی که انجام میدهد، بر اساس تابع e رابطه زیر خواهد شد

$$o(e^{lognloglogn)})$$

در این الگوریتم یک Public Key & Private Key ایجاد میشود، برای Public Key عدد اول P و برای Private Key عدد اول q به صورت تصادفی در نظر گرفته میشود. حاصلضرب pxq=n به طوری که تابع ø(N) برابر است با (p-1) x (q-1)

میتوانیم یک عدد به نام e که بین یک و ø(N) است را انتخاب کنیم، سپس مقدار بزرگترین مخرج مشترک طبق رابطه زیر به دست خواهد آمد که بین e و ø(1)، میشود یک. یعنی = gcd(e,ø(N))=1

این عدد e به عنوان کلید برای رمزگذاری مطرح میشود، حال اگر بخواهیم رمزبرداری را انجام دهیم، رابطه زیر را نشان میدهد e.d = 1 mod ø(N) در این رابطه d عددی است که بین صفر و N قرار دارد در نتیجه d برای رمزبرداری کردن این الگوریتم استفاده میشود.

با توجه به این دو کلید، کلید رمزگذاری KU و کلید رمزبرداری KI را در قالب مجموعه به صورت زیر تعریف میکنیم

KU={e,N}
KR={d,p,q}

کاربرد RSA: وقتی KU و KR به دست آمدند، پیام M را میتوانیم با کمک کلید عمومی، KU رمز کنیم این عمل طبق رابطه زیر انجام میشود.

$$C = m^e mod\ N$$

در این رابطه 0<M<N است.

برای رمزبرداری کردن از کلید Private یعنی KR استفاده میکنیم. در این صورت رابطه زیر نمایان گر رمزبرداری است.

$$M = C^d mod\ N$$

تذکر: پیام M باید از بلاک ماژول N کوچکتر باشد

RSA چرا و چگونه کار میکند:

با کمک و استفاده از فرمول اویلر و روابط ریاضی مربوط به آن.

در RSA دو عامل مهم وجود دارد، یکی N و دومی ø(N)

- because of Euler's Theorem:
- $a^{ø(n)} mod\ N = 1$
    - where gcd(a,N)=1

- in RSA have:
  - N=p.q
  - ø(N)=(p-1)(q-1)

در رابطه مربوط به پیدا کردن e و d انتخاب p و q به ما کمک میکند تا ø(N) را در اجرای زمانی الگوریتم با محاسبه بیشتر

$$e.d=1+k.ø(N)$$

اثبات فرمول را به صورت استقرایی میتوان در این رابطه دید:

hence :
$C^d = (M^e)^d = M^{1+k.ø(N)} = M^1.(M^{ø(N)})^q = M^1.(1)^q = M^1 = M \bmod N$

نمونه استفاده از RSA با اعداد تصادفی ۱۱ و ۱۷

1. Select primes: p=17 & q=11
2. Compute $n = pq$ =17×11=187
3. Compute ø(n)=(p–1)(q-1)=16×10=160
4. Select e *:* gcd(e,160)=1; choose e=7
5. Determine d*:* de=1 mod 160 and d < 160 Value is d=23 since 23×7=161= 10×160+1
6. Publish public key KU={7,187}
7. Keep secret private key KR={23,17,11}

و همچنین خواهیم داشت:

- sample RSA encryption/decryption is:
- given message M = 88 (nb. 88<187)
- encryption:

  $C = 88^7 \bmod 187 = 11$
- decryption:

  $M = 11^{23} \bmod 187 = 88$

تذکر: این الگوریتم بر اساس، ضرب های ماژولار و تقسیمات ماژولار از CPU زیادی استفاده میکند، شکل کلی الگوریتم به صورت زیر در یک حلقه for خواهد بود.

```
c ← 0; d ← 1
for i ← k downto 0
    do  c ← 2 × c
        d ← (d × d) mod n
        if  b_i = 1
            then  c ← c + 1
                  d ← (d × a) mod n
return d
```

قدرت امنیتی RSA:

به این نوع رمزگذاری سه نوع حمله انجام میشود

۱- Brute Force or Dictionary Attack
۲- حمله ریاضی Mathematical Attack که روی تابع Ø اثر دارد
۳- حمله در وقت اجرا یا timing Attack که در حال اجرا میتواند، ماشین رمز را از کار بیاندازد.

جلسه نهم

تحلیل DES: از آنجائیکه DES جز رمزهای بلوکی یا Feistel Cipher Structure میباشد و از n بیت، متن اولیه بلوک n بیتی رمز شده را بیرون میدهد. دو به توان n بلوک متن اولیه متفاوت وجود خواهد داشت و میتوان به صورت معکوس روی آنها یک متن رمز ایجاد کرد. در رمزهای بلوکی اصول زیر را خواهیم داشت.

- یک رمز جانشینی برگشت پذیر دلخواه، به ازای اندازه بلوک بزرگ امکان پذیر نیست.
- از نظر پیاده سازی و کارایی

- برای چنین تبدیلی، تناظر توسط مقادیر ستون دوم تعریف میشود.
- در جدول اسلاید قبل، تناظر توسط مقادیر ستون دوم تعریف میشود.
- در واقع، مقادیر متن رمز به ازای هر بلوک متن اصلی را نشان میدهد.
- این همان کلید است که یک تناظر خاص را از بین همه تناظر ها مشخص میکند
- در این حالت، اندازه کلید میشود: 4bit x 16 rows = 64 bit

- به ازای یک بلوک n بیتی، اندازه کلید n x n^2 خواهد بود.
- به ازای بلوک 64 بیتی (که اندازه خوبی برای پیشگیری از حملات آماری است)، جدولی با 2^64 سطر لازم داریم
- و اندازه کلید میشود 64 x 2^64 = 2^70 = 10^21 بیت

Plaintext	Ciphertext
0000	1110
0001	0100
0010	1101
0011	0001
0100	0010

جانشینی بلوکی عمومی N x N به ازای N = 4

در اصول رمزگذاری بلوکی، بهتر آن است که اندازه بلوک را کوچکتر کنیم و رمز بلوکی یک جانشین عمومی n بیتی از متن اصلی باشد.
در استانداردهای رمزنگاری DES چون از رمز ضربی و جایگشتی استفاده میشود و تعداد رند های DES زیاد میشود، این رمز سنگین خواهد شد و نیاز به CPU و RAM زیاد خواهد داشت. در روند کلی برای ورودی یک جایگشت انجام میشود و اگر ورودی ۶۴ بیت باشد، ۱۶ دور روی جایگشت ها انجام خواهد شد. یک جایگشت نهایی که معکوس جایگشت اول است خواهیم داشت و یک جایگشت اولیه روی کلید که به دو قسمت تقسیم شده است نیز وجود دارد.
در نهایت شانزده مرحله با کمک یک شیفت چرخشی به چپ و جایگشت دو نیمه روی زیرکلید ها انجام میشود.
در s-box نیز تعدادی confusion یا تداخل انجام میشود که s-box تداخل ها را پخش میکند. حال در تحلیل میتوان گفت که از نظر سرعت وجود s-box باعث میشود که رمز گذاری روی بیت های بیشتر از ۶۴ بیت ورودی به طور تصاعدی پایین بیاید.
بنابراین، هنگام رمزگشایی، اگر یک بیت در کلید یا داده ها تغییر کند، منجر به پدیده بهمنی میشود و تمام یا نصف بیت های خروجی نیز تغییر پیدا میکند. بنابراین در قدرت DES دو نگرانی وجود دارد.

۱- اندازه کلید
۲- طبیعت الگوریتم

در اندازه کلید، اگر کلید ۵۶ بیتی باشد، 7.2 x 10^16 مقدار متفاوت خواهیم داشت. در حالت دوم اگر s-box دارای هشت Decoder یا encoder باشد، رمزگشایی آن سخت تر خواهد بود. در روش حملات آماری، حمله کننده میتواند، کلید را به صورت آماری و با استفاده از تابع جستجوی فضای حالت و توزیع نرمال کلید را به دست آورد.

معمولا تحلیل های رمز روی DES، شامل تحلیل رمز دیفرانسیلی، تحلیل رمز خطی و حملات مرتبط با کلید می باشد.

در حملات زمانی که از طریق زمان رمز گشایی، میتواند، متن رمز شده را پیدا کند، الگوریتم DES قوی تر است.

در رمز خطی و دیفرانسیلی، عامل نفوذ به رمز و شکستن آن بلوک های n بیتی هستند و معمولا به صورت درختی، از سمت چپ به راست قابل Trace کردن میباشد.

تمرین: با توجه به تئوری Feistel بیان کنید که اثر بهمنی روی قدرت DES و استخراج کلید های آن چگونه و با چه رابطه ای تعریف میشود و S-Box ها چگونه طراحی میشوند. آیا برای ۱۲۸ بیت کلمه ورودی هشت S-Box کافی است یا خیر.

جلسه دهم

ویژگی های بنیادین سیستم های رمزنگاری متقارن

این رمز ها شامل مشخصه های عملیاتی زیر هستند:

۱- پراکنده هستند، یعنی از نظر شاخص های آماری ضریب همبستگی وجود ندارد و دارای ضریب پراکندگی بالاست. یعنی کل متن، رمز شده و پراکندگی رمز در کل متن بالا میرود.
۲- گمراه کننده هستند، یعنی بین ورودی و خروجی و کلید، هیچ رابطه ای پیدا نمیشود.
۳- اثر بهمنی (فروپاشی) کوچکترین تغییر در ورودی یا کلید بسیار گسترده و پیچیده خروجی را تغییر میدهد ولی روی میزان پراکندگی اول (قبل از تغییر) هیچ اثری ندارد.

این سیستم های رمزنگاری میتوانند در تولید رمز، طول ورودی و خروجی برابری را نشان میدهند. مدل سازی رمزنگار روی آنها امکان پذیر نیست. قدرت رمزنگاری روی کلید است و از قوانین زیر تبعیت میکند:

۱- سیستم رمزنگاری اگر نه به لحاظ تئوری که در عمل غیر قابل شکست است
۲- سیستم رمزنگاری باید هیچ گونه نکته پنهان و محرمانه ای نداشته باشد بلکه تنها چیزی که باید سری نگاه داشته شود کلید رمز است (اصل اساسی کرکهف)
۳- کلید رمز باید به گونه ای قابل انتخاب باشد که بتوان به راحتی آن را عوض کرد، همچنین بتوان آنرا به خاطر سپرد، نیازی به یادداشت نباشید
۴- متون رمزنگاری شده باید از طریق خطوط تلگراف قابل مخابره باشد
۵- دستگاه رمزنگاری یا اسناد رمزنگاری شده باید توسط یک نفر قابل حمل باشد
۶- دستگاه رمزنگاری باید به سهولت قابل راه اندازی و کاربردی باشد

قوانین ششگانه فوق را قوانین شهف (کرکهف) در رمزنگاری گویند. اصول رمزنگاری در این سیستم ها یا بر اساس تعویض است یا بر اساس جا به جایی و یا بر اساس اختصاصی بودن.

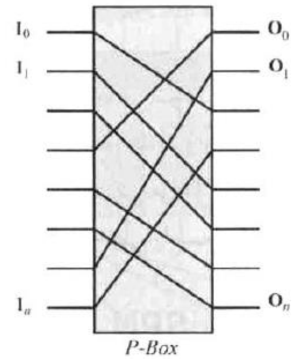
پروتکل های امنیتی دارای ویژگی های زیر است:
۱- نشست
۲- تازگی
۳- افزونگی

این پروتکل ها سه تعریف زیر را دارد و بر این اساس، دسته بندی میشود:

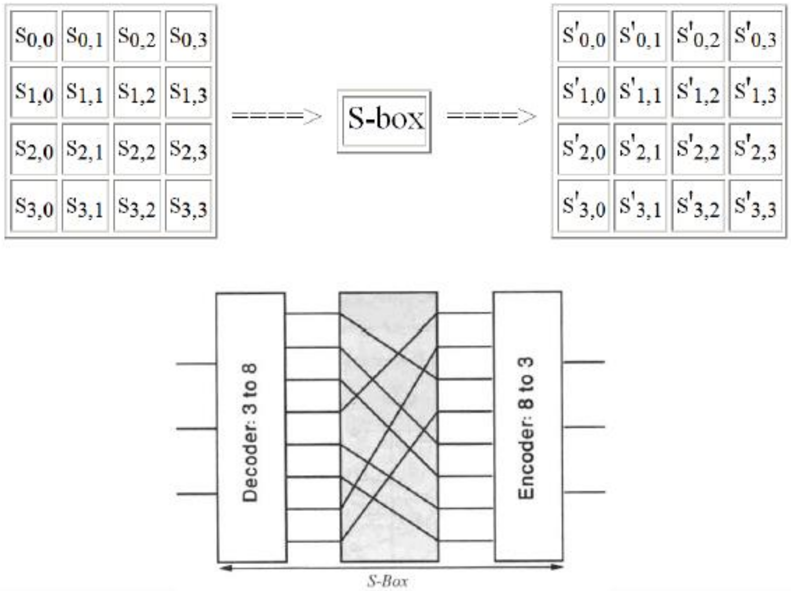
۱- پروتکل های امنیتی مبتنی بر حکمیت عنصر ثالث Arbitrated Protocols
۲- پروتکل های امنیتی مبتنی بر قضاوت Adjudicated Protocols
۳- پروتکل های خود اتکا self-enforced protocols

سیستم های رمزنگاری مدرن، دارای اجزای خاص میباشد که به شرح زیر است:

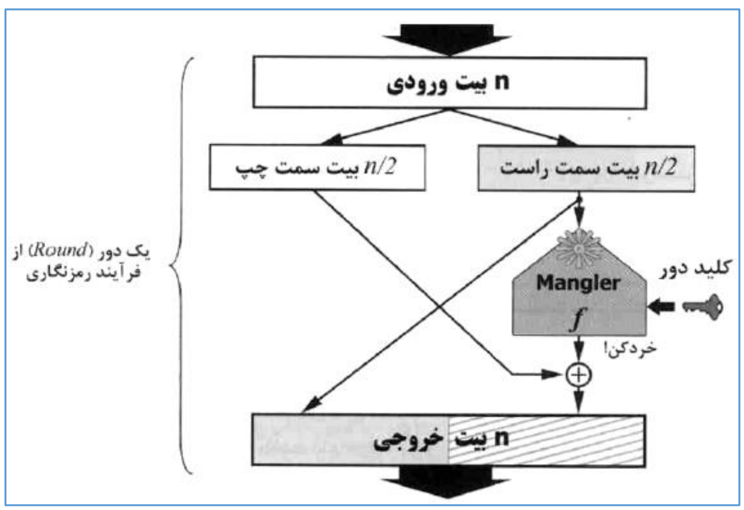
۱- جعبه تعویض یا p-box /// شکل ۱ ///
۲- جعبه جانشینی یا نگاشت یا s-box این جعبه از decoder ورودی میگیرد و به encoder خروجی میدهد /// شکل ۲ ////
۳- S-box مختلط که ترکیبی از s-box ها و p-box ها است. به طوری که p-box ها در ورودی و خروجی s-box ها قرار میگیرند، این روش افزایش خاصیت غیر خطی در عملیات رمزنگاری است و پیچیدگی الگوریتم رمزنگاری را بیشتر میکند. لازم به ذکر است شبکه ای از ترکیب s-box و p-box داشته باشیم و کلید ها را با هم در هر مرحله xor کنیم و نتیجه آن را به مرحله یا دور بعد بفرستیم.
۴- پیدایش معماری در رمزنگاری: با توجه به سه عامل قبل معماری در رمزنگاری تعریف شد و یکی از این معماری ها الگوریتم DES و معماری فیستل بود. /// شکل ۳ ///



شکل ۱



شکل ۲



شکل ۳

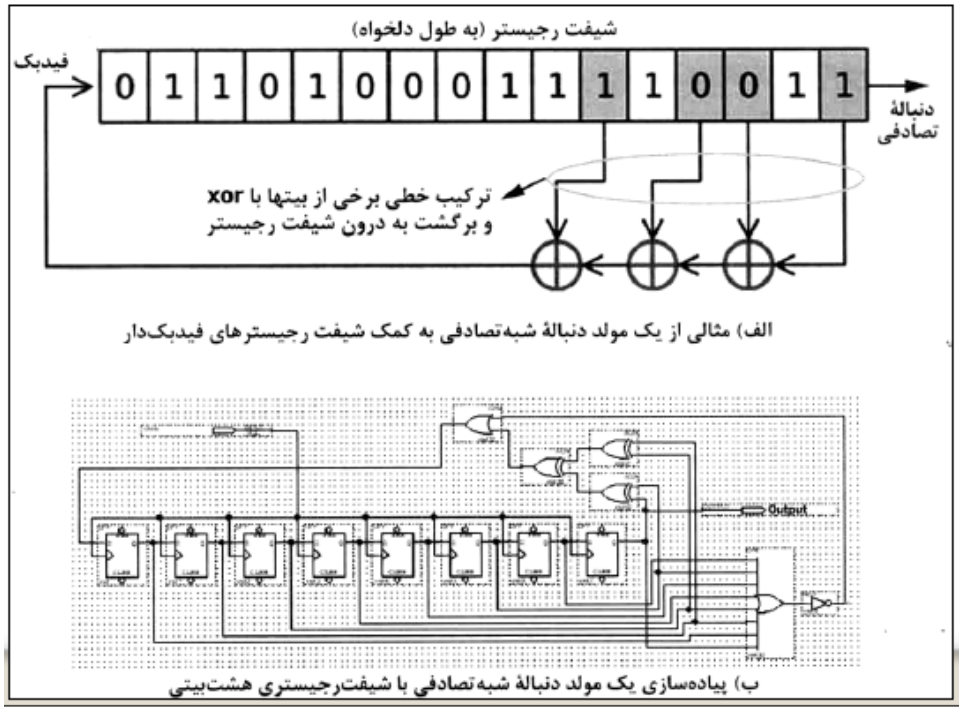
در این معماری ها معمولا شکست n بیت ورودی را داریم و یک جا به جایی یا جایگشت نیز روی شکست وجود خواهد داشت. این معماری ها معمولا کلید ها را با xor به هم متصل میکند و یک دور یا round از فرایند رمزنگاری را در هر مرحله به وجود می آورند. به همین دلیل است که الگوریتم های DES دارای دورهای متعدد رمزنگاری می باشد که در این دورها عمل xor و در خروجی آنها عمل جابه جایی انجام میشود.

در رمزگذاری ها میتوانیم از ماتریس و دنباله ها استفاده کنیم. که به آنها OTP گویند. OTP دارای شرایط زیر است

۱- دنباله بیت ها کاملا تصادفی باشند. (احتمال صفر و یک: ۰٫۵ / استقلال بیت های تولید شده)
۲- طول کلید و متن برابر باشند (کلید تکراری / قابل پیش بینی و عدم استقلال بیت ها)
۳- در رمزنگاری هیچگاه از یک کلید دوبار استفاده نشود.

OTP سختی های خود را نیز دارد، این سختی ها شامل روش های تولید اعداد تصادفی به کمک همنهشتی و الگوریتم shift register است.

$$X_{i+1} = (a.x_i + b) \bmod m$$



الف) مثالی از یک مولد دنبالهٔ شبه‌تصادفی به کمک شیفت رجیسترهای فیدبک‌دار

ب) پیاده‌سازی یک مولد دنبالهٔ شبه‌تصادفی با شیفت‌رجیستری هشت‌بیتی

پـــــایـــــان