
بسم الله الرحمن الرحيم

تمرین سوم درس امنیت

تمرین

مبینا کاشانیان

۱۳۹۹-۱۴۰۰

فهرست مطالب

۱ تحقیق

در این تمرین از ما خواسته شده بود که درباره ی دو موضوع به تحقیق بپردازیم. سوالات بیان شده برای تحقیق :

۱. در مورد حملات صورت گرفته به DES و نوع آن توضیح دهید؟

۲. با شکسته شدن الگوریتم DES ،در سال ۲۰۰۱ الگوریتم AES به عنوان استاندارد رمزنگاری انتخاب شد.در مورد نحوه این انتخاب، ساختار و چگونگی کارکرد این الگوریتم تحقیق کنید؟

حال به پردازش هر یک از موضوعات فوق میپردازیم.

۱.۰.۱ حملات DES

در این قسمت به توضیح ۶ مورد از حملاتی که میتوان برای الگوریتم DES به کار گرفت را بیان میکنیم. DES یک الگوریتم رمزنگاری متقارن است که در ژانویه ۱۹۷۷ به عنوان استانداردی برای حفاظت اطلاعات غیر طبقه‌بندی‌شده در ایالات‌متحده توسط اداره ملی استانداردهای ملی (که اکنون به عنوان موسسه ملی استانداردها و فن‌آوری شناخته می‌شود)استفاده شد. به طور گسترده برای حفاظت از اطلاعات حساس و تایید اعتبار معاملات بانکی به کار می‌رود. ما در اینجا شش روش مختلف برای شکستن DES را پیشنهاد می‌کنیم. در ابتدا این ۶ روش را بیان میکنیم و به شرح مختصری از آنها میپردازیم:

۱. Exhaustive key search

رمزنگاری ،DES الگوریتمی است که بلوک های داده ۶۴ بیتی را با استفاده از یک کلید رمز ۵۶ بیتی به کار می‌برد. یک سناریوی که میتوان برای شکستن این الگوریتم پیشنهاد داد آن است که ما یک بلوک رمزنگاری شده در اختیار داریم، ما اطلاعاتی در مورد text plain داریم (مثلا می‌دانیم که آن یک متن ASCII یا یک تصویر JPEG است) حال با این دانسته ما می‌خواهیم کلید رمز را بازیابی کنیم. روش ساده‌تر تلاش برای رمزگشایی بلوک با تمام کلیدهای ممکن است. اطلاعاتی که ما در text plain داریم به ما این امکان را می‌دهد که کلید صحیح را تشخیص دهیم و جستجو را متوقف کنیم. به طور متوسط، باید (۳۶میلیون میلیارد یا ۳۶ هزار تریلیون)کلید را امتحان کنیم. و تحقیقات نشان داده است که یک کامپیوتر شخصی معمولی می‌تواند هر ثانیه حدود یک یا دو میلیون کلید داشته باشد، این نشان‌دهنده زمان کاری حدود ۶۰۰ تا ۱۲۰۰ سال برای یک ماشین واحد است که در واقع این همان الگوریتم Brute force است.

۲. A dedicated machine

جستجوی جامع زمان بر روی یک کامپیوتر شخصی است بسیار زمان بر است، اما امکان پذیر است. در سال ۱۹۹۸، EFF یک

دستگاه است که با نام Deep Crack شناخته شد و از این جهت به وجود آمد که به جهان نشان دهد که DES یک الگوریتم ایمن نیست. این دستگاه می‌تواند کلید را با کمک یک جستجوی جامع در ۴ روز به طور متوسط بازیابی کند و ۹۲ میلیارد کلید را هر ثانیه بررسی کند.

۳. A huge cluster of computers

یکی از راه هایی است که نیاز به پول زیادی ندارد که DES را بشکند. . در ژانویه ۱۹۹۹ ،Distributed.Net سازمانی متخصص در جمع‌آوری و مدیریت زمان بیکاری کامپیوترها از داوطلبان درخواست کرد که در هنگامی که از کامپیوتر خود استفاده نمیکنند منابع کامپیوتر را برای انجام این ازمایش در اختیار بگذارند و در نهایت این سازمان، کلید DES را در ۲۳ ساعت شکست داد. و بیشتر از صد هزار کامپیوتر (از کند ترین تا قوی ترین کامپیوتر)، بخش کوچکی از کار را دریافت و انجام داده‌اند.

۴. A time-memory tradeoff

در حال حاضر این امکان وجود دارد که یک سناریو را تصور کنیم: حافظه موجود زیادی داریم و برای تمام کلیدهای ممکن k بلوک رمزنگاری شده y متناظر با یک بلوک معین x داده و ذخیره‌سازی جفت y و k آماده هستیم. بنابراین اگر در اختیار یک نسخه رمزنگاری شده x از بلوک معروف خود با یک کلید ناشناخته k را با جستجو در این نوع دیکشنری، در اختیار داشته باشیم، قادر خواهیم بود بسیار سریع کلید مناسب را پیدا کنیم. این روش در مواردی که بیش از یک کلید برای یافتن آن داریم، جالب می‌شود و ما حافظه کافی در اختیار داریم.
در سال ۱۹۸۰، مارتین هلمن در یک الگوریتم پیشنهاد کرد که به زمان کمتری نسبت به جستجوی جامع و حافظه کم‌تر نسبت به روش ذخیره‌سازی نیاز دارد. الگوریتم او به ترتیب ۱۰۰۰ گیگابایت امکانات ذخیره‌سازی و حدود ۵ روز محاسبات برای یک کامپیوتر شخصی نیاز دارد.

۵. Differential cryptanalysis

در سال ۱۹۹۰، دو جاسوس اسرائیلی که در موسسه Weitzmann کار می‌کردند، یک تکنیک عمومی جدید برای شکستن الگوریتم های متقارن به نام cryptanalysis differential را ابداع کردند. این اولین بار بود که یک روش می‌تواند DES را در کم‌تر از یک جستجوی جامع شکست دهد. تصور کنید که ما ابزاری داریم که داده‌ها را با یک کلید محرمانه متصل می‌کند و تصور می‌کند که ما ابزار لازم برای "خواندن" کلید در چیپ ها را نداریم. کاری که ما می‌توانیم انجام دهیم این است که چند بلوک از داده‌ها را انتخاب کنیم و آن‌ها را با دستگاه رمز کنیم. یک مزیت بزرگ این حمله این است که احتمال موفقیت آن به صورت خطی با تعداد احتمالات موجود برای test plain انتخاب‌شده افزایش می‌یابد و در نتیجه حتی با plaintexts منتخب کمتری نیز انجام می‌شود.

۶. Linear cryptanalysis

یک روش کلی مهم دیگر برای شکستن رمزها روش رمزهای خطی است توسط یک محقق ژاپنی اختراع شد، امکان بازیابی کلید متناظر در چند روز با استفاده از cryptanalysis خطی وجود دارد. این مهم‌ترین حمله به DES است که در این زمان شناخته شده‌است. یک پروژه تحقیقاتی حاضر به نام LASEC تحلیل هزینه این حمله است که تحقیقات نشان داده است

با ۱۸ cpu در حال اجرای کلید DES در ۴ روز است.هدف از این پروژه انجام یک آنالیز آماری بهتر بر پیچیدگی آن و احتمال موفقیت آن است

DES دیگر نمی‌تواند به عنوان یک الگوریتم رمزنگاری ایمن در نظر گرفته شود. سازمان NIST یک فرآیند را برای توسعه یک استاندارد جدید، موسوم به AES (استاندارد رمزگذاری پیشرفته)راه‌اندازی کرده‌است، که DES را برای ۱۰ سال آینده جایگزین خواهد کرد.

منبع : DES Attack

۲.۰.۱ الگوریتم AES

من در قسمت بعدی که پیاده سازی همین الگوریتم را داشتیم توضیحاتی را ارائه کردم. اما در اینجا نیز به صورت مختصر الگوریتم را شرح میدهم. همان طور که در قسمت قبلی ذکر کردم ۶ راه برای شکستنن الگوریتم رمزگذاری DES به وجود آمد و برای یک الگوریتمی که قرار است رمزگذاری بکند و امنیت را ایجاد کند همچنین الگوریتمی شکست کامل و ضعف محسوب میشود پس بنابراین باید یک راه بهتر برای رمزگذاری انتخاب میشد. همان طور که گفتم الگوریتم DES از یک کلید ۵۶ بیتی استفاده میکرد ولی AES اینگونه عمل نکرد و کلید های متفاوت ۱۲۸و۱۹۲ و ۲۵۶ بیتی را جایگزین کرد وبلاک سایز ها هم ۱۲۸ بیتی بودند به همین دلیل بسیار قدرتمند تر از DES به حساب آمد. AES شامل یک سری از گام‌های جانشینی و جایگشت برای ایجاد بلوک رمز شده می‌باشد. در فصل بعدی کامل کارکرد این الگوریتم را شرح کردم.

۲ گزارش تمرین عملی

۱.۲ الگوریتم AES

در این تمرین از ما خواسته شده بود که الگوریتم AES را پیاده سازی کنیم و همان طور که میدانیم این الگوریتم بر اساس شبکه جانشینی جایگشت طراحی شده است و این الگوریتم بر روی آرایه‌های ۴*۴ بایتی با ترتیب ستونی به نام حالت عمل می‌کند که در واقع یک ماتریس از داده های ورودی ایجاد میشود و در این الگوریتم اندازه‌ی کلید الگوریتم تعداد دورهایی را که طی آن ورودی را به خروجی رمز شده تبدیل می‌شود را مشخص می‌کند و این تعداد دفعات برای کلید ۱۲۸ بیت ۱۰ دور، کلید ۱۹۲ بیتی ۱۲ دور و کلید ۲۵۶ بیتی ۱۴ دور است ، از طرفی هم هر چه تعداد دور ها بیشتر شود پیچیدگی آن رمز بیشتر میشود به این معنی که الگوریتم AES با کلید ۲۵۶ بیتی و با ۱۴ دور قوی ترین میباشد. الگوریتم AES چهار مرحله کلی است. در مرحله اول کلیدهای چرخه با استفاده از زمان‌بندی کلید AES به دست می‌آید. در مرحله‌ی بعد با کمک یک XOR بیتی بایت هر حالت با بایت کلید دور ترکیب می‌شود. مرحله سوم خود ۴ بخش دارد و بسته به نوع الگوریتم (۱۲۸، ۱۹۲ و یا ۲۵۶ بیتی) ۹، ۱۱ و یا ۱۳ بار تکرار خواهد شد. بخش اول این مرحله SubBytes است که در آن بر اساس یک جدول جستجو بایت‌ها به صورت غیر خطی جابه‌جا می‌شوند پس گام SubBytes نوبت به مرحله ی ShiftRows است که در آن حداقل سه سطح از حالت به تعداد گام‌های معینی شیفت پیدا می‌کند.بخش سوم مرحله‌ی چرخه‌ها بخش MixColumns خواهد بود. در این بخش ۴ بایت از هر ستون حالت با یک انتقال خطی ترکیب می‌شوند. این تابع چهار بایت به عنوان ورودی دریافت کرده و ۴ بایت خروجی تولید می‌کند. این بخش به همراه بخش قبل پخش شدن رمزنگاری را فراهم می‌کند.بخش آخر مرحله‌ی سوم گام AddRoundKey است. در این بخش زیر کلید با حالت ترکیب می‌شود. برای هر دور، یک زیرکلید از کلید اصلی با کمک زمان‌بندی کلید ریندال ایجاد شده و هر زیرکلید هم اندازه‌ی حالت خواهد بود. زیرکلید با ترکیب هر بایت از حالت با بایت متناظرش در زیرکلید به کمک xor بیتی جمع می‌شود.در مرحله‌ی نهایی، گامهای ShiftRows SubBytes، و addRoundKey یک دور دیگر اجرا شده و تعداد کل دورهای اجرای الگوریتم را به ۱۰، ۱۲ و یا ۱۴ دور می‌رسانند.

۲.۲ پیاده سازی

همان طور که در بالا بیان شد که الگوریتم AES چیست و بر چه اساسی کار میکند، در این قسمت پیاده سازی الگوریتم را بیان میکنیم. الگوریتمی که پیاده سازی آن را شرح میدهم AES-۲۵۶ است و برای اینکار ابتدا کتابخانه pip install pycryptodomex

را توسط pip نصب میکنیم.

Encryption ۱.۲.۲

الگوریتم AES-۲۵۶ برای داده های خود نیازمند به ۱۶ بایت بلوک میباشد و این وابسته به مودی هست که الگوریتم پیاده سازی میشود برای مثال در مود GCM نیازی نیست که پدینگ توسط ما شکل بگیرد. حال در ابتدا تابع encrypt را پیاده سازی میکنیم و این فانکشن دو ورودی میگیرد یکی Plain Text و دیگری Password حال توسط این تابع از پسورد استفاده میکنیم تا text plain را رمز گذاری کنیم بنابراین هر کس با داشتن encrypted text و پسورد میتواند آن را رمزگشایی کند. در ابتدا عدد رندوم بر حسب بایت را تولید میکنیم در واقع این عدد رندوم به ما این امکان را میدهد که برای هر فرایند تکرار شونده ی encryption یک عدد جدید تولید شود و در واقع با اینکار چون هر دفعه یک عدد رندوم جدید تولید میشود کار حمله کننده بسیار سخت میشود چون پیدا کردن عدد رندوم در دنیای نامحدود اعداد بسیار سخت است و سایز های بلاک را ۱۶ میگذاریم سپس فرایند تولید کلید را انجام میدهیم در اینجا از kdf scrypt استفاده میکنیم که به ما این امکان را میدهید که پسورد های جدیدی بر اساس پسورد های قبلی بسازیم در واقع از این طریق یک key private از پسورد تولید میکنیم که با این کار برای فرد حمله کننده سخت خواهد شد که از طریق الگوریتم brute force رمزگشایی کند. پارامتر های مورد استفاده در scrypt به ترتیب N،R،P هستند که درواقع N باید از توان ۲ باشد و هرچقدر مقدارش بیشتر باشد امنیت کلید بیشتر خواهد شد. R منظور سایز بلاک میباشد. P برای مشخص کردن عملیات به صورت پارالل است و اینکه روی چند هسته اجرا شود. داده‌های نوع بایت را به رشته۶۴base کدگذاری میکنیم و برای هر یک tag تعریف میکنیم که این برچسب برای تایید داده‌ها هنگام استفاده از AES در حالت gcm استفاده می‌شود. این تضمین می‌کند که هیچ‌کس نمی‌تواند داده‌های ما را بدون اینکه ما در مورد آن مطلع شویم، تغییر دهد و یک nonce با مقدار دلخواه و باید یک مقدار تصادفی و منحصر به فرد برای هر بار تولید میشود که تابع encryption با همان کلید استفاده می‌شود. مثل یک نمک تصادفی برای یک رمز است. کتابخانه ما را با یک nonce امن تامین می‌کند.

Decryption ۲.۲.۲

همان طور که گفته شد با داشتن encrypted text و پسورد میتوانیم متن را رمزگشایی کند. پس به عنوان ورودی این تابع متن encrypt شده را با همان پسورد میدهیم و از طرفی از تابع encypt مقدایر رندوم را داریم به عبارت دیگر تابع decryption را برای رمزگشایی پیاده سازی میکنیم که به همان عدد تصادفی و nonce و برچسبی که برای رمزنگاری استفاده کردیم نیاز دارد. ما از یک دیکشنری برای راحتی در تجزیه استفاده کردیم که تمامی پارامتر های مورد نیاز را از آن استخراج میکنیم ، پارامترهای مورد استفاده در توابع Scrypt و AES نیز باید به همان پارامتر هایی باشد که در تابع رمز نگاری استفاده شدند.

Main ۳.۲.۲

در قسمت main برنامه با ابتدا ورودی پسورد و متنی که کاربر میخواهد کد کند را دریافت میکنیم و سپس خروجی تابع encode شده را با هم مشاهده میکنیم و درنهایت هم به متن اولیه که کاربر وارد کرده بود می رسیم.