به نام خداوند بخشنده‌ی مهربان

دانشگاه صنعتی شریف | تمرین سری سوم* | دانشکده‌ی مهندسی کامپیوتر

مدرس: مهدی خرازی | امنیّت داده و شبکه | خزان ۱۳۹۸

اهداف تمرین

- آشنایی با حمله CRIME
- آشنایی با حمله‌ی مردِ میانی
- آشنایی با پروتکل Diffie-Hellman
- آشنایی با ARP Poisoning

۱. مقدمه

این تمرین شامل دو بخش است که هر کدام ۵۰ درصد نمره کل تمرین را شامل می‌شود.

در بخش اول با حمله CRIME آشنا می‌شوید و باید با استفاده از این روش به پرچمی [۱] که در کوکی [۲] مرورگر توسط کارگزار[۳] رمز شده است، دست پیدا کنید.

در بخش دوم لازم است حمله‌ی مردِ میانی[۴] یا به اختصار MitM را پیاده‌سازی کنید و پیام‌های رمزشده‌ای که بین کارگزار و کارخواه[۵] رد و بدل می‌شود را شنود کنید.

---

*با سپاس از افرا امینی، فاطمه حسنی و سولماز سلیمی.

[1] flag
[2] cookie
[3] server
[4] Man In the Middle
[5] client

۲. بخش اول ـ حمله‌ی CRIME[۶]

۱.۲ پیش‌نیازها

برای آشنایی با این آسیب‌پذیری باید ابتدا با نحوه کلی عملکرد الگوریتم‌های فشرده‌سازی آشنا شویم. الگوریتم‌های فشرده‌سازی مانند gzip به‌طور کلی از دو ترفند برای فشرده کردن پرونده‌ها استفاده می‌کنند:

- کلمات پرتکرارتر در نهایت به نمایشی با طول کوتاه‌تر در نسخه فشرده‌شده تبدیل می‌شوند.
- هر عبارتی که تکرار شود در نهایت تنها یک بار ذخیره می‌شود.

در توضیح مورد دوم باید گفت اگر کلمه‌ای در متن تکرار شود تنها بار اول ذخیره می‌شود. بار دوم و دفعات بعدی تنها اشاره‌گری به محل ذخیره شدن آن کلمه نگه‌داری می‌شود. برای مشخص‌تر شدن موضوع فرض کنید می‌خواهیم این عبارت را فشرده کنیم:

بوستان بر سرو دارد آن نگار دلستان
آن نگار دلستان، بر سرو دارد بوستان

در این صورت نحوه نگه‌داری فشرده‌شده این متن به کمک اشاره‌گرها به شکل زیر خواهد بود:

یعنی مصراع دوم این بیت پس از فشرده‌سازی تنها از سه اشاره‌گر و یک ویرگول تشکیل شده‌است.

۲.۲ توضیح مسأله[۷]

در کلاس درس با الگوریتم AES[۸] برای رمزنگاری پرونده‌ها آشنا شده‌اید. در این تمرین خواهید دید که رمزنگاری اطلاعات ارزشمند به تنهایی، آن‌ها را امن نمی‌کند. بعد از دریافت اطلاعات شنود شده یکی از ابتدایی‌ترین اطلاعاتی که می‌توان بدست آورد تعداد بایت‌های منتقل شده است. واضح است که رمزنگاری از درز این اطلاعات ساده جلوگیری نمی‌کند. حال اگر بدانیم این اطلاعات پیش از رمزشدن فشرده شده‌اند با اطلاع از نحوه عملکرد الگوریتم‌های فشرده‌سازی که درقسمت قبل توضیح داده شد، می‌توان به اطلاعات رمزشده دست پیدا کرد. با توجه به آن‌چه گفته شد می‌دانیم که اگر دورشته ورودی پیش از فشرده‌سازی طول یکسان داشته باشند، بعد از فشرده‌سازی رشته‌ای طول کمتری پیدا می‌کند که کاراکترهای تکراری بیشتری داشته باشد. بنابراین یک حمله می‌تواند به این

[6] https://en.wikipedia.org/wiki/CRIME
[7] compression side channel attack
[8] Advanced Encryption Standard

شکل باشد که با حدس و خطا کاراکترهایی را پیدا کنیم که اگر به رشته فشرده‌شده اضافه شوند کمترین افزایش طول را به همراه داشته باشند.

۳.۲ پیاده‌سازی

با رفتن به اینجا یک پیام ساده دریافت می‌کنید. به این‌صورت که اگر اسم خود را در پارامترهای get این url وارد کرده باشید پیام «Hello [name]» را می‌بینید که name همان اسم وارد شده است و در غیر این‌صورت پیام به صورت «Hello guest» نمایش داده می‌شود. علاوه بر این یک کوکی هم توسط این کارگزار در مرورگر شما تنظیم می‌شود. برای راحتی می‌توانید از این دستور استفاده کنید:

```
curl -vk https://pacific-anchorage-60533.herokuapp.com/ce442/?user=MyName
```

اگر به ساختار کوکی دقت کنید، متوجه می‌شوید که در آن یک کلید پرچم وجود دارد که محتوای آن رمز شده است. وظیفه شما این است که این پرچم را بدست آورید. کد اجرا شده در کارگزار به شکل زیر است:

```
def get_auth(user):
  with open('valuable_data/flag.txt') as content_file:
    flag = content_file.read()
  data = [user, flag]
  json_text = json.dumps(data)
  zip = zlib.compress(json_text.encode('ascii'))
  backend = default_backend()
  key = os.urandom(32)
  iv = os.urandom(16)
  cipher = Cipher(algorithms.AES(key), modes.CTR(iv), backend=backend)
  encryptor = cipher.encryptor()
  ct = encryptor.update(zip) + encryptor.finalize()
  return base64.b64encode(ct)

def process_req(request):
  if request.method == 'GET':
    user = request.GET.get("user", "guest")
    resp = JsonResponse({"message" : "Hello " + user})
    resp.set_cookie("flag", get_auth(user))
    return resp
```

همان‌طور که در کد کارگزار مشاهده می‌کنید، کارگزار پس از دریافت درخواست get نام کاربر را به تابع `get_auth` ارسال می‌کند (در صورتی که اسم وارد نشده باشد،عبارت guest را می‌ارسال می‌کند). این تابع پرچم را از پرونده‌ی حاوی عبارت پرچم می‌خواند. پرچم یک عبارت به طول ۱۵ با قالب زیر است:

```
flag:<10 BYTE STRING OF THESE CHARS: a-z A-Z>
```

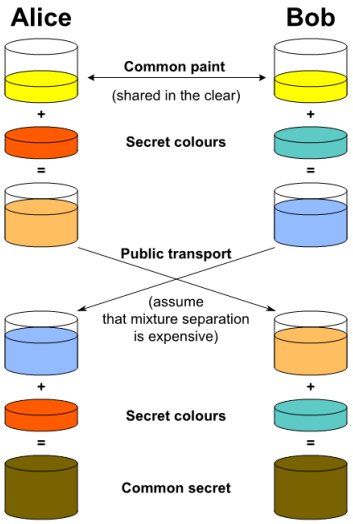
سپس عبارت پرچم را به همراه اسم کاربر فشرده می‌کند. در ادامه از الگوریتم AES برای رمزنگاری داده فشرده‌شده در مرحله قبل استفاده کرده و درنهایت عبارت رمزشده‌ی حاصل را برمی‌گرداند. مقدار برگردانده شده که عبارت رمزشده‌ی پرچم و نام کاربر است به عنوان کوکی در مرورگر کاربر تنظیم می‌شود. وظیفه شما این است که با سعی و خطا و استفاده از مطالب گفته شده در مورد الگوریتم فشرده‌سازی gzip ، پرچمی را که در flag.txt قرار داده شده به دست آورید.

۳. بخش دوم- حمله‌ی مردِ میانی

۱.۳ پیش‌نیازها

۱.۱.۳ پروتکل Diffie-Hellman [۹]

این پروتکل جهت رد و بدل کردنِ امن یک کلید خصوصی بین دو نفر (یا گروه) کاربرد دارد. فرض کنید Alice می‌خواهد با Bob بر سر یک کلید خصوصی به توافق برسد تا از این به بعد بتوانند بقیه پیام‌ها را با این کلید خصوصی رمز کنند و فقط خودشان بتوانند آن‌ها را رمزگشایی کنند. شکل۱ ایده اصلی این پروتکل را به تصویر می‌کشد.



شکل ۱: ایده‌ی اصلی پروتکل Diffie-Hellman (منبع تصویر)

مراحل پروتکل به شرح زیر است:

۱. Alice و Bob توافق می‌کنند که از پایه g و پیمانه p استفاده کنند. پایه و پیمانه عمومی است و همه آن را می‌دانند.

۲. Alice یک عدد دلخواه a انتخاب می‌کند و $A = g^a \mod p$ را برای Bob می‌فرستد.

۳. Bob هم یک عدد دلخواه b انتخاب می‌کند و $B = g^b \mod p$ را برای Alice می‌فرستد.

۴. Alice مقدارِ $s = B^a \mod p$ را محاسبه می‌کند و از آن به عنوان کلید خصوصی استفاده می‌کند.

۵. Bob مقدارِ $s = A^b \mod p$ را محاسبه می‌کند و از آن به عنوان کلید خصوصی استفاده می‌کند.

[۹] https://en.wikipedia.org/wiki/Diffie-Hellman_key_exchange

توجه داشته باشید که رابطه‌ی زیر برقرار است:

$$(g^a \mod p)^b \mod p = g^{ab} \mod p$$
$$(g^b \mod p)^a \mod p = g^{ab} \mod p$$

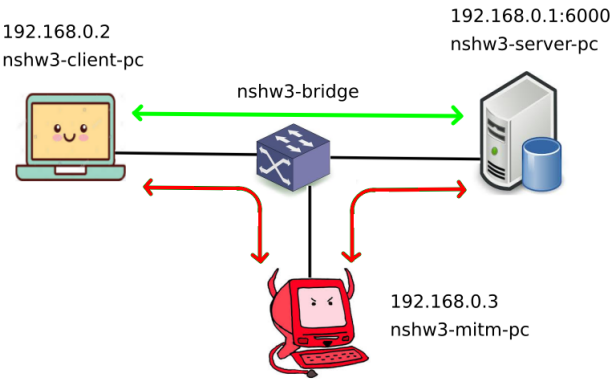
و بنابراین هر دو سمت این پروتکل روی یک کلید مشترک به توافق خواهند رسید.

۲.۱.۳ ARP Poisoning [10]

این تکنیک در شروع بسیاری از حمله‌ها مورد استفاده قرار می‌گیرد و به این صورت است که حمله‌کننده بسته‌هایی از نوع پاسخ ARP را مکرراً در شبکه می‌فرستد و آدرس های IP که متعلق به او نیست را با آدرس MAC خود به شبکه داخلی معرفی می‌کند. به این ترتیب بسته‌ها به جای مقصد اصلی به حمله‌کننده تحویل داده می‌شوند و حال او می‌تواند بسته‌ها را مشاهده کند، تغییر دهد یا دور بریزد.

۲.۳ توضیح مسأله

درون ماشین مجازی که در اختیار شما قرار می‌گیرد، شبکه ای مانند شکل۲ شبیه سازی شده است. کارخواه با آدرس 192.168.0.2 به کارگزار با آدرس 192.168.0.1 و پورت ۶۰۰۰ اتصال TCP برقرار می‌کند و با استفاده از پروتکل Diffie-Hellman به تبادل کلید[11] می‌پردازند. از آن پس پیام‌های رمزشده‌ای بین کارخواه و کارگزار رد و بدل می‌شود که در نهایت در یکی از این پیام‌ها می‌توان پرچم را یافت. این روند چند ثانیه یک بار از ابتدا تکرار می‌شود. شما به عنوان حمله‌کننده باید در میان این ارتباط قرار بگیرید و پرچم را بدست آورید.



شکل ۲: شبکه‌ی مجازی شامل حمله‌کننده، کارگزار و کارخواه

[10] https://en.wikipedia.org/wiki/ARP_spoofing
[11] key exchange

۱.۲.۳. جزئیات رمزنگاری

رمزنگاری پیام‌ها با استفاده از AES-128 در حالت CBC انجام می‌شود. به این منظور مقدارِ SHA256 کلید مشترک را (که با پروتکل Diffie-Hellman مبادله شده است) در حالت Big Endian حساب می‌کنیم، سپس مقدارِ MD5 آن را بدست می‌آوریم تا کلیدی با طول مناسب برای رمزنگاری AES تولید شود. توجه کنید که برای SHA256 مقدارِ digest و برای MD5 مقدارِ hexdigest باید محاسبه شود. در نهایت بعد از رمز کردن پیام، مقدار IV تصادفی را در ۱۶ بایت اول قرار می‌دهیم و کل محتوا را در مبنای ۶۴[۱۲] کد و ارسال می کنیم.

۳.۳. پیاده سازی

ماشین مجازی را از اینجا بارگیری و نصب کنید (اگر VirtualBox بر روی سامانه‌ی خود ندارید، ابتدا آن را نصب کنید و سپس پرونده‌ی ova. دریافت شده را import کنید). نام کاربری و کلمه‌ی‌عبور به شرح زیر است:

```
username: ubuntu
password: ubuntu
```

اگر خواستید با ssh به ماشین مجازی متصل شوید، پورت مقصد را ۲۲۲۲ قرار دهید:

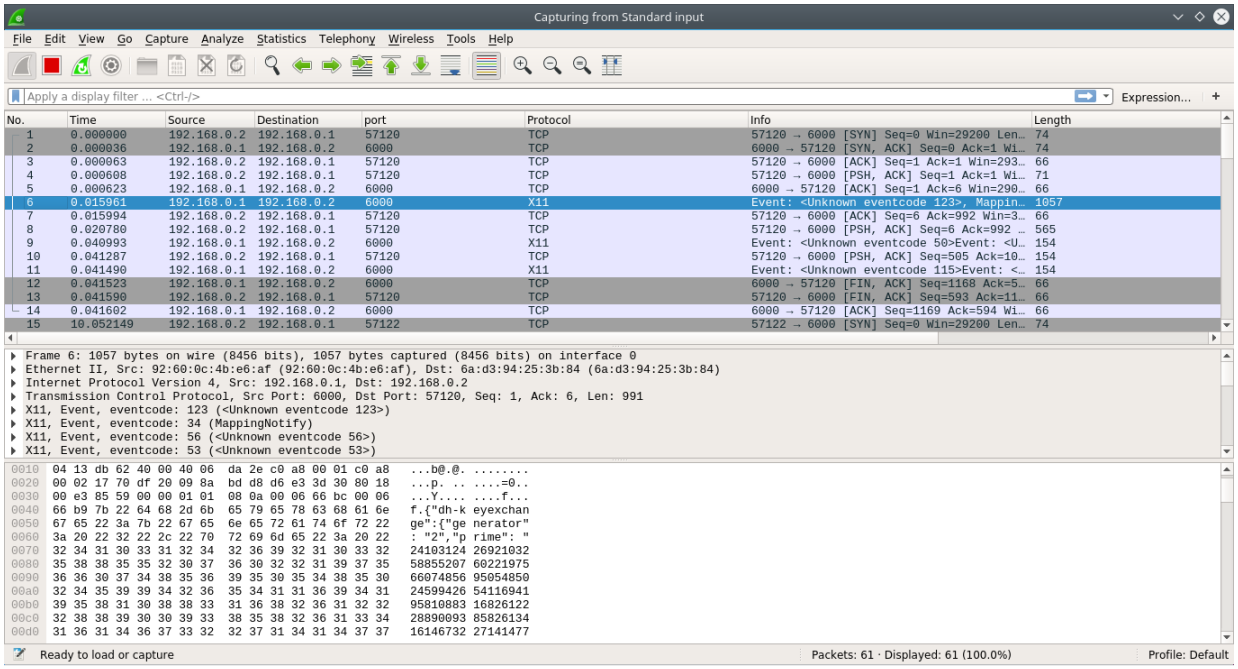
```
ssh -p 2222 ubuntu@127.0.0.1
```

بعد از روشن شدن سیستم کد کارخواه و کارگزار شروع به اجرا می‌کنند. برای مشاهده ترافیک بین آن ها، ابزار wireshark را بر روی سیستم خودتان (میزبان) نصب کنید و دستورات زیر را وارد کنید (سیستم عامل میزبان linux فرض شده است، در صورتی که از ویندوز به عنوان ماشین میزبان استفاده می‌کنید می‌توانید نحوه‌ی ضبط کردن ترافیک ماشین مجازی از طریق میزبان و wireshark را جست‌وجو کنید):

```
ssh-copy-id -p 2222 ubuntu@127.0.0.1
ssh -p 2222 ubuntu@127.0.0.1 "tcpdump -s 0 -i nshw3-bridge -w -" | sudo
 wireshark -k -i -
```

نمونه‌ی آنچه مشاهده خواهید کرد در شکل۳ آمده است. برای شروع حمله، سعی کنید با استفاده از ARP Poisoning ترافیک بین کارگزار و کارخواه را به سمت خود جذب کنید. سپس سعی کنید با مشاهده بسته‌ها بفهمید در چه قالبی با هم صحبت می کنند و داده های مورد نیاز برای تبادل کلید را ارسال می‌کنند (در این حالت می‌توانید IP Forwarding را فعال کنید تا ارتباط بین کارگزار و کارخواه قطع نشود). البته محتوای بسته ها را در محیط wireshark نیز می‌توانید ببینید. وقتی مطمئن شدید در مسیر عبور بسته‌ها قرار گرفته‌اید، محتوای آن‌ها را تغییر دهید و حمله را کامل کنید. در نهایت کد شما باید مقدار پرچم را خروجی دهد.
به نکات زیر توجه کنید:

[12] base64



شکل ۳: بسته های رد و بدل شده بین کارگزار و کارخواه در محیط wireshark

- قالب پرچم به شکل `flag{[an md5 digest]}` است.
- شما با رابط[۱۳] nshw3-mitm-pc و آدرس 192.168.0.3 به سوئیچ nshw3-bridge متصل هستید و نیازی به تعامل با رابط‌های دیگر ندارید.
- طول تمام پیام‌ها به اندازه‌ای است که از یک بسته IP فراتر نمی‌رود.
- دقت کنید که اگر طول بسته‌ها را در حین حمله تغییر دهید، پروتکل TCP دچار مشکل خواهد شد.
- کارگزار و کارخواه از قبل کلید مشترکی در اختیار دارند که به وسیله‌ی آن کارخواه معمایی را که کارگزار طرح می‌کند پاسخ می‌دهد. کارگزار تنها بعد از دریافت پاسخ صحیح، پرچم را ارسال می کند. بنابراین شما نمی‌توانید تنها با اتصال به یک طرف ارتباط پرچم را بدست بیاورید و باید حمله‌ی مرد میانی را به طور کامل پیاده سازی کنید.
- برای پیاده سازی پروتکل Diffie-Hellman و همچنین رمزنگاری AES می‌توانید از کتابخانه‌های موجود استفاده کنید.
- در پروتکل Diffie-Hellman پارامتر g را تنها از مقادیر $\{2, 3, 5, 7\}$ و پارامتر p را بین ۱۲۸ تا ۱۹۲ بایت انتخاب کنید.

[13] interface

- شما مجاز به استفاده از هر زبانی هستید. پیشنهاد می‌کنیم از پایتون و کتابخانه‌ی scapy برای تغییر بسته‌ها استفاده کنید.

- می‌توانید قسمت checksum را در سرآیندهای IP و TCP بسته‌های ارسالی‌تان از بین ببرید تا مشکلی در تغییر محتوای بسته‌ها بوجود نیاید.

- اگر لازم داشتید ابزار خاصی نصب کنید، بدون نیاز به گذرواژه و با دستور `sudo apt-get install` می‌توانید آن را دریافت کنید.

- اگر از زبانی به غیر از پایتون استفاده می‌کنید، با استفاده از دستور زیر قابلیت[۱۴] NET_RAW را به فایل اجرایی یا مفسر [۱۵] زبان خود بدهید:

```
sudo addnetcap /PATH_TO_EXECUTABLE_OR_INTERPRETER
```

خطوط زیر مثال هایی برای زبان جاوا و c++ را نشان می‌دهد:

```
sudo addnetcap /usr/bin/java
sudo addnetcap /home/ns/a.out
```

- پیشنهاد می‌کنیم برای عیب‌یابی کد خود، به همان صورت که پیش‌تر توضیح داده شد، از ابزار wireshark استفاده کنید. همچنین درون ماشین مجازی ابزار tshark نصب شده است که می‌توانید از آن نیز بهره ببرید.

[14] capability
[15] interpreter

۴.۳. کد های کمکی

بعد از بدست آوردن کلید مشترک Diffie-Hellman با استفاده از تابع زیر مقدار درهم‌شده‌ی[۱۶] آن را محاسبه کنید، سپس باقی مراحل را با این مقدار جدید ادامه دهید. (کد زیر برای پایتون ۲ و ۳ کار می‌کند.)

```
import hashlib

def get_key(dhKey):
    bs = []
    while dhKey != 0:
        bs.append(dhKey & 0xFF)
        dhKey >>= 8
    shared_secret_bytes = bytes(bytearray(reversed(bs)))
    s = hashlib.sha256()
    s.update(bytes(shared_secret_bytes))
    return s.digest()
```

تابع زیر که در سرور و کلاینت استفاده می‌شود و برای راهنمایی به شما داده شده نیز خروجی تابع بالا را به عنوان کلید دریافت می‌کند:

```
def aes_encrypt(string, key):
    aeskey = md5(key).hexdigest()
    iv = Random.new().read(AES.block_size)
    cipher = AES.new(aeskey, AES.MODE_CBC, iv)
    return b64encode(iv + cipher.encrypt(string.ecnode()))
```

برای عملیات decrypt هم نیاز است تابعی مشابه بنویسید.

[16] hashed

۴. تحویل دادنی ها

برای ارسال نهایی تمرین نیاز دارید تا مانند تمرین های قبل یک پوشه به نام hw3 در مخزن خود ایجاد نمایید و موارد زیر را درون آن قرار دهید.

- گزارش: تمام فعالیت‌های خود را در گزارشی با نام report.pdf به صورت کامل بنویسید.
- پرچم: به ازای هر بخش شما باید یک عبارت پرچم بدست آورید و در گزارش خود وارد کنید.
- اسکریپت حمله: همه‌ی کدهای مورد استفاده‌ی خود را که در گزارش به آن‌ها اشاره کرده‌اید در پوشه‌ای به اسم `codes` و در زیرپوشه‌های `part1` و `part2` مربوط به هر بخش تمرین قرار دهید.

۵. نکات ضروری

- تمام برنامه‌ی شما باید توسط خود شما نوشته شده باشد. فرستادن کل یا قسمتی از برنامه‌تان برای افراد دیگر، یا استفاده از کل یا قسمتی از برنامه‌ی فرد دیگری، حتی با ذکر منبع، تقلب محسوب می‌شود.
- ارسال پاسخ و راهنمایی در گروه‌های تلگرام و سایر منابع عمومی به منزله تقلب محسوب خواهد شد.
- در صورتی‌که هر مشکل یا پرسشی داشتید که فکر می‌کنید پاسخ آن برای همه مفید خواهدبود، آن را در فهرست پستی (میلینگ لیست) ارسال نمایید.
- از فرستادن جواب تمرین به فهرست پستی خودداری کنید.
- دقت کنید که پس از انجام این تمرین ساختار نهایی مخزن شما به شکل زیر باشد:

```
├──README.md
├──hw3/
│  ├──codes/
│  │  ├──part1
│  │  └──part2
│  └──report.pdf
```

- همه‌ی پرونده‌های لازم را با همان نامی که در این مستند ذکر شده است، با دستورهای زیر ارسال کنید (فرض شده مخزن خود را در مسیر home قرار داده‌اید):

```
cd ~/ce441-981-student_id/hw3
```

```
git status
```

```
git add --all
```

```
git commit -m "Finished my third assignment"
```

```
git push origin master
```