



اهداف تمرین

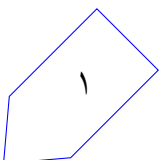
- آشنایی با حمله CRIME
- آشنایی با حمله مردِ میانی
- آشنایی با پروتکل Diffie-Hellman
- آشنایی با ARP Poisoning

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 رمز شده‌ای که بین کارگزار و کارخواه^۵ رد و بدل می‌شود را شنود کنید.

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

- ^۱flag
- ^۲cookie
- ^۳server
- ^۴Man In the Middle
- ^۵client



۲. بخش اول - 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 اطلاع ساده جلوگیری نمی‌کند. حال اگر بدانیم این اطلاعات پیش از رمز شدن فشرده‌شده‌اند با اطلاع از نحوه عملکرد الگوریتم‌های فشرده‌سازی که در قسمت قبل توضیح داده‌شد، می‌توان به اطلاعات رمز شده 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

^۶<https://en.wikipedia.org/wiki/CRIME>

^۷compression side channel attack

^۸Advanced Encryption Standard

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

۳.۲. پیاده‌سازی

با رفتن به اینجا یک پیام ساده دریافت می‌کنید. به این صورت که اگر اسم خود را در پارامترهای get این url وارد کرده باشید پیام «Hello name» را می‌بینید که همان اسم وارد شده است و در غیر این صورت 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:

```
1 def get_auth(user):
2     with open('valuable_data/flag.txt') as content_file:
3         flag = content_file.read()
4     data = [user, flag]
5     json_text = json.dumps(data)
6     zip = zlib.compress(json_text.encode('ascii'))
7     backend = default_backend()
8     key = os.urandom(32)
9     iv = os.urandom(16)
10    cipher = Cipher(algorithms.AES(key), modes.CTR(iv), backend=backend)
11    encryptor = cipher.encryptor()
12    ct = encryptor.update(zip) + encryptor.finalize()
13    return base64.b64encode(ct)
14
15 def process_req(request):
16     if request.method == 'GET':
17         user = request.GET.get("user", "guest")
18         resp = JsonResponse({"message": "Hello " + user})
19         resp.set_cookie("flag", get_auth(user))
20     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، آن را به همراه اسم کاربر فشرده می‌کند. سپس از الگوریتم AES برای رمزنگاری داده فشرده‌شده در مرحله قبل استفاده کرده و در نهایت 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`، به پرچمی که در مرورگر تنظیم می‌شود را به دست آورید.

در این بخش تمرین، هدف پیدا کردن پرچمی با قالب زیر است:

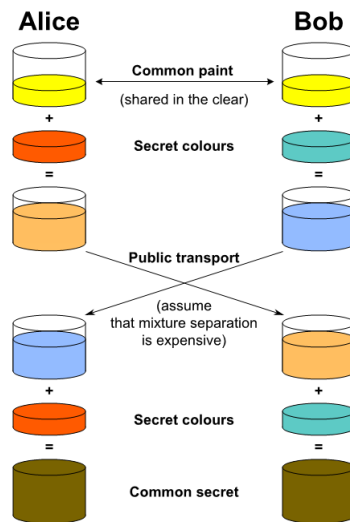
```
flag:<10 BYTE STRING OF THESE CHARS: a-z A-Z>
```

۳. بخش دوم - حمله‌ی مردِ میانی

۱.۳. پیش‌نیازها

۱.۱.۳. پروتکل Diffie-Hellman^۹

این پروتکل جهت رد و بدل کردنِ امنِ یک کلید خصوصی بین دو نفر (یا گروه) کاربرد دارد. فرض کنید Alice می‌خواهد با Bob بر سر یک کلید خصوصی توافق کند تا از این به بعد بتوانند بقیه پیام‌ها را با این کلید خصوصی رمز کنند و فقط خودشان بتوانند آن‌ها را رمزگشایی کند. شکل ۱ 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 پیمانانه p استفاده کنند. پایه و پیمانانه عمومی است و همه آن‌را می‌دانند.

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

۳. Bob هم یک عدد دلخواه b انتخاب می‌کند و $B = g^b \pmod{p}$ را به Alice می‌فرستد.

۴. Alice مقدار $s = B^a \pmod{p}$ را محاسبه می‌کند و از آن به عنوان کلید خصوصی استفاده می‌کند.

۵. Bob مقدار $s = A^b \pmod{p}$ را محاسبه می‌کند و از آن به عنوان کلید خصوصی استفاده می‌کند.

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

توجه داشته باشید که رابطه‌ی زیر برقرار است:

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

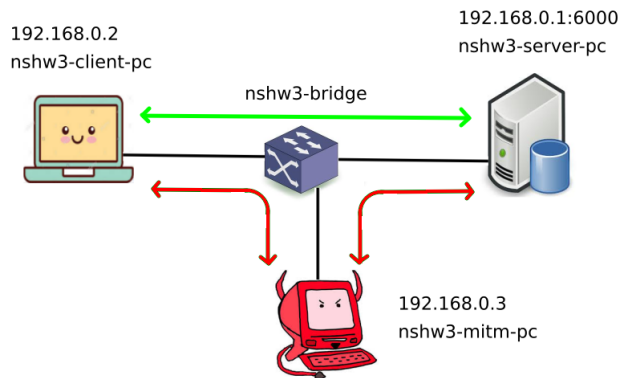
$$(g^b \bmod p)^a \bmod p = g^{ab} \bmod p$$

۲.۱.۳. ۱° ARP Poisoning

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 ارسال می‌کند و آدرس‌های 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، شبکه‌ای مانند شکل ۲ شبیه‌سازی 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^{۱۱} 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

۱.۲.۳. جزئیات رمزنگاری

رمزنگاری پیام‌ها با استفاده از AES-128 در حالت CBC انجام می‌شود. به این منظور مقدار md5 کلید مشترک را (که با پروتکل Diffie-Hellman مبادله شده است) حساب می‌کنیم تا کلیدی با طول مناسب برای رمزنگاری AES

^{۱۰}https://en.wikipedia.org/wiki/ARP_spoofing

^{۱۱}key exchange

تولید شود. در نهایت بعد از رمز کردن پیام، آن را در مبنای ۱۲۶۴ 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 کلمه‌ی عبور 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 آن‌ها، 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 -
```

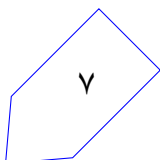
نمونه‌ی آنچه مشاهده خواهید کرد در شکل ۳ آمده است.

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 می‌کنند و داده‌های 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 بسته‌ها 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:

• قالب پرچم به شکل `flag{MD5}` است.

base64^{۱۲}



No.	Time	Source	Destination	port	Protocol	Info	Length
1	0.000000	192.168.0.2	192.168.0.1	57120	TCP	57120 → 6000 [SYN] Seq=0 Win=29200 Len...	74
2	0.000036	192.168.0.1	192.168.0.2	6000	TCP	6000 → 57120 [SYN, ACK] Seq=9 Ack=1 Wi...	74
3	0.000063	192.168.0.2	192.168.0.1	57120	TCP	57120 → 6000 [ACK] Seq=1 Ack=6 Win=293...	66
4	0.000608	192.168.0.2	192.168.0.1	57120	TCP	57120 → 6000 [PSH, ACK] Seq=1 Ack=1 Wi...	71
5	0.000623	192.168.0.1	192.168.0.2	6000	TCP	6000 → 57120 [ACK] Seq=1 Ack=6 Win=290...	66
6	0.015961	192.168.0.1	192.168.0.2	6000	X11	Event: <Unknown eventcode 123>, Mappin...	1057
7	0.015994	192.168.0.2	192.168.0.1	57120	TCP	57120 → 6000 [ACK] Seq=6 Ack=992 Win=3...	66
8	0.020780	192.168.0.2	192.168.0.1	57120	TCP	57120 → 6000 [PSH, ACK] Seq=6 Ack=992...	565
9	0.040993	192.168.0.1	192.168.0.2	6000	X11	Event: <Unknown eventcode 50>Event: <U...	154
10	0.041287	192.168.0.2	192.168.0.1	57120	TCP	57120 → 6000 [PSH, ACK] Seq=505 Ack=10...	154
11	0.041490	192.168.0.1	192.168.0.2	6000	X11	Event: <Unknown eventcode 115>Event: <...	154
12	0.041523	192.168.0.1	192.168.0.2	6000	TCP	6000 → 57120 [FIN, ACK] Seq=1168 Ack=5...	66
13	0.041590	192.168.0.2	192.168.0.1	57120	TCP	57120 → 6000 [FIN, ACK] Seq=593 Ack=11...	66
14	0.041602	192.168.0.1	192.168.0.2	6000	TCP	6000 → 57120 [ACK] Seq=1169 Ack=594 Wi...	66
15	16.052149	192.168.0.2	192.168.0.1	57122	TCP	57122 → 6000 [SYN] Seq=0 Win=29200 Len...	74

▶ Frame 6: 1057 bytes on wire (8456 bits), 1057 bytes captured (8456 bits) on interface 0
 ▶ Ethernet II, Src: 92:60:0c:4b:e6:af (92:60:0c:4b:e6:af), Dst: 6a:d3:94:25:3b:84 (6a:d3:94:25:3b:84)
 ▶ Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.2
 ▶ Transmission Control Protocol, Src Port: 6000, Dst Port: 57120, Seq: 1, Ack: 6, Len: 991
 ▶ X11, Event, eventcode: 123 (<Unknown eventcode 123>)
 ▶ X11, Event, eventcode: 34 (MappingNotify)
 ▶ X11, Event, eventcode: 56 (<Unknown eventcode 56>)
 ▶ X11, Event, eventcode: 53 (<Unknown eventcode 53>)

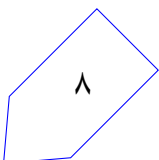
0010 04 13 db 62 40 00 40 06 da 2e c0 a8 00 01 c0 a8 ...b0@.....
 0020 00 02 17 78 df 20 09 8a bd d8 d6 e3 3d 30 80 18 ...p... ..=0..
 0030 00 e3 85 59 00 00 01 01 08 0a 00 06 66 bc 00 06 ...Y.... ..f...
 0040 66 b9 7b 22 64 68 2d 6b 65 79 65 78 63 68 61 6e f.{"dh-k eyexchan
 0050 67 65 22 3a 7b 22 67 65 6e 65 72 61 74 6f 72 22 ge":{"ge nerator"
 0060 3a 20 22 32 22 c2 22 70 72 69 6d 65 22 3a 20 22 : "2", "p rime": "
 0070 32 34 31 30 33 31 32 34 32 36 39 32 31 30 33 32 24103124 26921032
 0080 35 38 38 35 35 32 30 37 36 30 32 32 31 39 37 35 58855207 60221975
 0090 36 36 30 37 34 38 35 36 39 35 30 35 34 38 35 30 66074856 95054850
 00a0 32 34 35 39 39 34 32 36 35 34 31 31 36 39 34 31 24599426 54116941
 00b0 39 35 38 31 30 38 38 33 31 36 38 32 36 31 32 32 95810883 16826122
 00c0 32 38 38 39 30 30 39 33 38 35 38 32 36 31 33 34 28890093 85826134
 00d0 31 36 31 34 36 37 33 32 32 37 31 34 31 34 37 37 16146732 27141477

Packets: 61 · Displayed: 61 (100.0%) Profile: Default

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

- شما با رابط ^{۱۳}nshw3-mitm-pc و آدرس 192.168.0.3 به سوئیچ nshw3-bridge متصل هستید و نیازی به تعامل با رابط های دیگر ندارید.
- طول تمام پیام ها به اندازه ای است که از یک بسته IP فراتر نمی رود.
- دقت کنید که اگر طول بسته ها را در حین حمله تغییر دهید، پروتکل TCP دچار مشکل خواهد شد.
- کارگزار و کارخواه از قبل کلید مشترکی در اختیار دارند که به وسیله ای آن کارخواه معمایی را که کارگزار طرح می کند پاسخ می دهد. کارگزار تنها بعد از دریافت پاسخ صحیح، پرچم را ارسال می کند. بنابراین شما نمی توانید تنها با اتصال به یک طرف ارتباط پرچم را بدست بیاورید و باید حمله ای مرد میانی را به طور کامل پیاده سازی کنید.
- برای پیاده سازی پروتکل Diffie-Hellman و همچنین رمزنگاری AES می توانید از کتابخانه های 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.
- شما مجاز به استفاده از هر زبانی هستید. پیشنهاد می کنیم از پایتون و کتابخانه ای scapy برای تغییر بسته ها استفاده کنید.

^{۱۳}interface



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

- بنابراین 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:

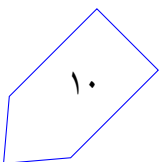
```
def AESencrypt(string, key):
    aeskey = md5(key).hexdigest()
```

^{۱۴}capability

^{۱۵}interpreter

^{۱۶}Hash

```
iv = Random.new().read(AES.block_size)
cipher = AES.new(aeskey, AES.MODE_CBC, iv)
return b64encode(iv + cipher.encrypt(string.encode()))
```



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 فعالیت های 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: همه ی کدهای مورد استفاده ی خود را که در گزارش به آنها اشاره کرده اید در پوشه ای به اسم codes و در زیرپوشه های part1 و part2 مربوط به هر بخش تمرین قرار دهید.

۵. نکات ضروری

- تمام برنامه ی شما باید توسط خود شما نوشته شده باشد. فرستادن کل یا قسمتی از برنامه تان برای افراد دیگر، یا استفاده از کل یا قسمتی از برنامه ی فرد دیگری، حتی با ذکر منبع، تقلب محسوب می شود.
- ارسال پاسخ و راهنمایی در گروه های تلگرام و سایر منابع عمومی به منزله تقلب محسوب خواهد شد.
- در صورتی که هر مشکل یا پرسشی داشتید که فکر می کنید پاسخ آن برای همه مفید خواهد بود، 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:

```
1 --README.md
2 --hw3/
3   --codes/
4     --part1
5     --part2
6   --report.pdf
```

- همه ی پرونده های 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 داده اید):

```
cd ~/ce442-971-student_id/hw3
```

```
git status
```

```
git add - -all
```

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

```
git push origin master
```

