



Assignment - A02

Webserver and HTTP Message Inspection

Penulis : AAT, AVA
Versi : 2 (20240903-1000)



© 2024, Fakultas Ilmu Komputer Universitas Indonesia
This work is licensed under [Creative Commons Attribution-ShareAlike 4.0 International \(CC BY-SA 4.0\)](https://creativecommons.org/licenses/by-sa/4.0/) license.

Riwayat Versi

Versi	Timestamp	Halaman	Perubahan
1	20240902-0800	Semua	Rilis Pertama
2	20240903-1000	5, 19 dan 22	Steps Retrieving Long Documents dan Penamaan Berkas
3	20240911-2100	17 dan 18	Screenshot IP Lokal dan IP Mesin Eksternal
4	20250912-0900	19 dan 20	Steps Retrieving Long Documents

Daftar Isi

Riwayat Versi	2
Daftar Isi	3
Informasi Umum	5
Ekspektasi Hasil Pembelajaran	5
Prasyarat.....	5
Google Cloud Platform Instance	5
Wireshark	5
Deskripsi.....	5
Docker Engine	5
Step 1: Memastikan Software Sudah Terupdate.....	6
Step 2: Menginstall Docker Engine	6
Step 3: Memberikan Akses ke Docker Engine.....	7
Step 4: Relogin	7
Step 5: Memastikan Docker Sudah Terinstall.....	8
Step 6: Melakukan Pull pada Docker Image	8
Step 7: Deployment pada Web Server	9
Step 8: Memastikan Web Server Berjalan dengan Baik	10
Step 9: Daftar Docker Container Aktif	11
Step 10: Mematikan Docker Container.....	12
Step 11: Menghapus Docker Container.....	12
HTTP/2 dan Wireshark.....	13
Step 1: Membuat Pre-Master Secret Key	13
Khusus Pengguna Windows.....	13
Khusus Pengguna Linux atau Mac	14
Memastikan Pre-Master Secret Key Sudah Dibuat.....	14
Step 2: Konfigurasi Wireshark	14
Notes	16
Spesifikasi.....	17
Steps.....	17
Basic HTTP Analysis	18

HTTP Content Typing	18
Content with Embedded Objects	20
Persistent HTTP (Caching)	20
[37 Points] Basic HTTP Analysis	21
[14 Points] HTTP Content Typing	21
[14 Points] Retrieving Long Documents	21
[14 Points] Content with Embedded Objects	21
[21 Points] Persistent HTTP (Caching)	22
Informasi Terkait Pengumpulan Berkas	22
Peraturan	23
Keterlambatan	23
Plagiarisme	23

Assignment – A02

Web Server and HTTP Message Inspection

Informasi Umum

- Tipe Tugas : Individu
- Batas Waktu Pengumpulan : Jumat, 13 September 2024 Jam 17.00 Waktu SCeLE
- Format Penamaan Berkas :
 - Laporan : A02_[NPM].pdf
 - Berkas Wireshark : A02_[NPM]_[KodeBerkas].pcapng
(Lihat bagian Informasi Pengumpulan Berkas)
- *Template* Lembar Jawaban : [Klik Di Sini](#)

Ekspektasi Hasil Pembelajaran

Mahasiswa diharapkan untuk menginspek (C3) HTTP message menggunakan Wireshark.

Prasyarat

Google Cloud Platform Instance

Dalam assignment ini, Anda akan men-*deploy* web server untuk memeriksa HTTP message. Langkah-langkah untuk men-*deploy* web server akan diberikan dalam dokumen ini. Sebelum memulai assignment ini, Anda harus menyiapkan instance GCP baru sesuai dengan spesifikasi yang dinyatakan dalam A01a: Pengantar Google Cloud Platform. Anda dapat memilih salah satu dari kedua mesin untuk mengerjakan *assignment* ini.

Wireshark

Anda harus memiliki Wireshark di PC / Laptop Anda sendiri. Untuk menyelesaikan *assignment* ini, Anda setidaknya harus memahami dasar-dasar Wireshark untuk analisis paket. Jika Anda belum menginstal Wireshark atau tidak terbiasa dengan aplikasi ini, pelajari kembali assignment sebelumnya.

Deskripsi

Docker Engine

Agar pengguna bisa mengunjungi website Anda, Anda perlu mendeploy website Anda. Ada banyak metode dalam melakukan *deployment*, seperti menggunakan Heroku. Namun, dalam *assignment* ini, Anda akan menggunakan instance GCP serta Docker untuk melakukannya.

Docker adalah *platform open source* untuk mengembangkan, mengirim, dan menjalankan aplikasi. Docker memungkinkan Anda untuk memisahkan aplikasi Anda dari infrastrukturnya. Pada dasarnya, Docker adalah platform virtualisasi yang memungkinkan *software* (dalam hal ini, web server) ditempatkan dalam *container* yang akan dikemas dan dieksekusi dalam lingkungan yang terisolasi. Platform ini memungkinkan Anda membuat Docker Image siap pakai yang akan menjadi *template* bagi Docker Container untuk melakukan *deploy*.

Tim Jarkom x Jarkomdat telah menyediakan Docker Image yang berisi template web server sederhana yang dibuat untuk tugas ini. Dalam *assignment* ini, kami akan memandu kalian untuk melakukan instalasi Docker dan mendapatkan Docker Image sebagai template untuk men-*deploy* web server kalian sendiri di instance GCP.



Step 1: Memastikan Software Sudah Terupdate

Anda disarankan untuk memastikan bahwa software yang telah terinstal pada instance GCP telah terupdate. Langkah ini tidak wajib, tetapi merupakan *best practice* untuk memastikan *software* yang ada telah terupdate ke versi terbaru.

```
sudo apt update && sudo apt upgrade -y
```

Step 2: Menginstall Docker Engine

Docker Engine menjalankan Container yang akan ditempatkan dalam instance GCP. Ada beberapa cara untuk menginstal Docker Engine, tetapi Docker telah menyediakan script yang berisi seluruh installation command. Dalam assignment ini, Anda akan menggunakan script tersebut untuk menginstal Docker. Anda dapat mencoba menginstal Docker menggunakan cara lain di sini (tidak ada poin bonus).

Script instalasi Docker dihosting di <https://get.docker.com> dan dapat diakses melalui web browser jika Anda ingin melihat cara kerjanya. Untuk menginstal Docker, Anda dapat mendownload script dan menjalankannya langsung di instance GCP Anda sendiri. Anda dapat menjalankan script dengan option '--dry-run' saat menjalankan script untuk mempelajari langkah-langkah script yang akan dijalankan saat dipanggil:

```
curl -fsSL https://get.docker.com -o get-docker.sh  
sudo sh ./get-docker.sh
```

```
dvd@a02-webserver:~$ exit
logout
Connection to 35.193.234.134 closed.

C:\Users\david\Desktop>ssh -i gcp-key dvd@35.193.234.134
Linux a02-webserver 5.10.0-19-cloud-amd64 #1 SMP Debian 5.10.149-2 (2022-10-21) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Jan 30 10:13:12 2023 from 103.171.147.121
dvd@a02-webserver:~$
```

Script dibuat untuk berjalan tanpa pengawasan, jadi Anda hanya perlu menunggu sampai script selesai dijalankan. Perintah yang sedang dijalankan akan ditampilkan di terminal.

Step 3: Memberikan Akses ke Docker Engine

Untuk tujuan keamanan, tidak semua user diizinkan untuk mengakses Docker Engine secara langsung. Secara default, hanya root user dan escalated user (melalui sudo) yang dapat mengakses Docker Engine. Docker menyediakan user group yang dapat mengakses Docker Engine.

Untuk mendaftarkan user saat ini ke user group tersebut, jalankan command berikut (perhatikan bahwa \$USER adalah variabel shell yang menyimpan nama user saat ini):

```
sudo usermod -aG docker $USER
```

```
dvd@a02-webserver:~$ sudo usermod -aG docker $USER
```

Step 4: Relogin

Pendaftaran pengguna ke grup pengguna Docker tidak berlaku secara langsung. Jika Anda menggunakan SSH, tutup koneksi SSH saat ini dan masuk kembali. Anda dapat menggunakan perintah `exit` untuk menghentikan koneksi SSH saat ini.

```
exit
```

```
dvd@a02-webserver:~$ exit
logout
Connection to 35.193.234.134 closed.

C:\Users\david\Desktop>ssh -i gcp-key dvd@35.193.234.134
Linux a02-webserver 5.10.0-19-cloud-amd64 #1 SMP Debian 5.10.149-2 (2022-10-21) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Jan 30 10:13:12 2023 from 103.171.147.121
dvd@a02-webserver:~$
```

Step 5: Memastikan Docker Sudah Terinstall

Setelah user saat ini terdaftar dengan benar di user group docker, user tersebut sekarang dapat dengan bebas mengakses Docker Engine tanpa memerlukan privilege escalation atau menggunakan root user. Untuk memverifikasi penginstalan Docker Engine dan user saat ini memiliki akses terhadapnya, jalankan perintah berikut:

```
docker -v
```

```
dvd@a02-webserver:~$ docker -v
Docker version 20.10.23, build 7155243
dvd@a02-webserver:~$
```

Step 6: Melakukan Pull pada Docker Image

Docker Image adalah template yang akan Anda gunakan untuk menyebarkan aplikasi Anda. Anda dapat membuat Docker Image Anda sendiri dari kode aplikasi dan mendeploynya dengan Docker. Namun untuk tugas ini, kami telah menyediakan Docker Image sehingga Anda hanya perlu melakukan pull.

Secara default, docker melakukan pull dari Docker Hub. Ada beberapa provider Docker Image lainnya, seperti AWS Elastic Container Registry dan GCP Container Registry. Docker Image yang akan digunakan dipublikasikan melalui Docker Hub dan diidentifikasi dengan nama compnetcsui/webserver-lab-http2:latest. Command `docker pull` akan mendownload versi terbaru (atau versi yang telah ditentukan) dari Image tersebut dan menyimpannya di mesin. Command berikut digunakan untuk melakukan pull pada Docker Image dari Docker Hub:

```
docker pull <image name>:<tag>
# Dalam kasus ini: docker pull compnetcsui/webserver-lab-http2:latest
```



```
mir@a02-webserver:~$ docker pull compnetcsui/webserver-lab-http2:latest
latest: Pulling from compnetcsui/webserver-lab-http2
7264a8db6415: Pull complete
33fbadb7f8c6: Pull complete
5558af53d261: Pull complete
3ae8a7119b2e: Pull complete
6aa810d264db: Pull complete
d2bd4257a0bd: Pull complete
7487d36af5fb: Pull complete
d565ab59a09c: Pull complete
Digest: sha256:5db09e9cd07e99de3af29883505f6e4efa44b59a396807fb48efb51a46406034
Status: Downloaded newer image for compnetcsui/webserver-lab-http2:latest
docker.io/compnetcsui/webserver-lab-http2:latest
mir@a02-webserver:~$
```

Step 7: Deployment pada Web Server

Pada sebelumnya, Docker telah mendapatkan image untuk web server yang disediakan oleh kami. Pada langkah ini, Anda akan menjalankan beberapa command untuk membuat container dan menjalankan web server di dalamnya.

```
docker run --name webserver-lab-http2 --restart <restart policy> -e ID=<your npm> -p 443:8443 -d <image name>
```

```
aat@webserver-a02:~$ docker run --name webserver-lab-http2 --restart unless-stopped
-e ID=1234567890 -p 443:8443 -d compnetcsui/webserver-lab-http2:latest
f9c982e3a517d6e6f81df99af96e5b1b7777065203edafd17587048a34ec2f41
aat@webserver-a02:~$
```

Ada beberapa option yang digunakan dalam penyebaran web server ini:

1. **--name** **<identifier>**
Option ini digunakan untuk memberikan unique identifier ke container yang Anda jalankan. Hal ini demi menghindari masalah umum dalam menjalankan banyak container dari image yang sama dalam satu mesin.
2. **--restart** **<restart policy>**
Docker menyediakan option restart untuk menentukan apakah container Anda dimulai secara otomatis saat container tersebut keluar, atau saat Docker direstart. Berikut adalah nilai yang mungkin dari option '--restart':
 - **no**: Jangan merestart container secara otomatis (default).
 - **on-failure[:max-retries]**: Restart container jika keluar karena error (exit code selain 0). Anda juga dapat membatasi berapa kali daemon Docker mencoba untuk merestart container menggunakan opsi 'max-retries' secara opsional.
 - **always**: Selalu restart container jika berhenti. Jika container dihentikan secara manual, container direstart hanya ketika Docker daemon direstart atau container itu sendiri direstart secara manual.

- **unless-stopped:** Mirip dengan 'always', kecuali ketika container dihentikan (baik secara manual ataupun tidak), container tidak direstart bahkan setelah Docker daemon direstart. Ini adalah option yang paling direkomendasikan untuk image ini.
3. **-e** **<VAR=VALUE>:** Environment Variables
Option ini digunakan untuk memasukkan environment variable dengan nama variabel VAR dan nilai VALUE ke dalam container. Dalam hal ini, environment variable ID akan diberikan nilai NPM Anda sendiri.
 4. **-p** **<HostPort>:<ContainerPort>:** Port Forwarding
Secara default, saat Anda menjalankan container menggunakan 'docker run', container tidak membuka port apa pun ke luar. Untuk membuat port dapat diakses dari luar, atau dari container Docker lainnya yang berjalan pada network berbeda, gunakan flag '--publish' atau '-p'. Flag ini membuat aturan firewall yang memetakan port pada host Docker (instance GCP) ke port container. Perintah -p 443:8443 akan meneruskan koneksi apa pun yang dibuat ke port 443(HTTPS) milik host (instance GCP) ke port 8443 web server dalam container.
 5. **-d:** Detached Mode
Dalam detached mode, container dijalankan di background dan id milik container akan diprint. Container yang dimulai dalam mode ini keluar ketika root process yang digunakan untuk menjalankan container keluar. Namun, jika option ini tidak digunakan, eksekusi container akan mengendalikan shell saat ini, sehingga apabila shell ditutup, maka container juga akan keluar. Jika kontainer memiliki interactive shell, maka option ini seharusnya dihilangkan. Namun, web server yang kita gunakan dimaksudkan untuk dijalankan secara otomatis.

Step 8: Memastikan Web Server Berjalan dengan Baik

Dalam instance GCP ini, Anda dapat memastikan apakah web server Anda telah berjalan dengan baik dengan memeriksa apakah koneksi yang dibuat ke port 443 instance GCP (termasuk koneksi lokal) diteruskan ke container. Anda dapat membuat HTTP request ke localhost (127.0.0.1 di sebagian besar mesin Linux) dan seharusnya, HTTP request tersebut akan dilayani oleh container Docker. Jika container sudah berjalan dengan baik, maka Anda akan menerima HTTP response.

```
curl https://127.0.0.1/<NPM>/short
```

Catatan: nilai NPM pada parameter URL harus sesuai dengan nilai variabel ID yang ditulis saat image dijalankan. Selain itu, karena certificate pada webserver ini self-signed, maka Anda perlu menggunakan argumen `-k``.

```
wir@a02-webserver:~$ curl -k https://127.0.0.1/1234567890/short
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Short | CSUI CompNet</title>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
  </head>
  <body>
    <h1>Short - Hello, 1234567890</h1>
    <p>
      Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod
      tempor incididunt ut labore et dolore magna aliqua. Nulla facilisi nullam
      vehicula ipsum a arcu cursus vitae congue. Neque egestas congue quisque
      egestas diam in arcu cursus. Vivamus at augue eget arcu. Nibh praesent
      tristique magna sit amet. Diam vel quam elementum pulvinar etiam non quam.
      Mauris sit amet massa vitae. Volutpat diam ut venenatis tellus in metus
      vulputate. Tellus at urna condimentum mattis pellentesque id nibh. Natus
      et malesuada fames ac turpis egestas integer eget. Viverra suspendisse
      potenti nullam ac tortor vitae purus faucibus. Augue eget arcu dictum
      varius dui at consectetur lorem. Sit amet facilisis magna etiam.
      Ullamcorper a lacus vestibulum sed. Enim facilisis gravida neque
      convallis.
    </p>
  </body>
</html>
wir@a02-webserver:~$
```

Untuk memastikan web server dapat diakses secara public, Anda dapat menggunakan komputer Anda untuk mengirim HTTP request ke external IP address instance GCP melalui web browser. Abaikan apabila Anda mendapatkan insecure connection warning. Hal ini terjadi akibat certificate yang digunakan untuk web server tersebut adalah self-checked certificate, bukan certificate yang didapatkan dari trusted third-party certificate authority (Anda akan mempelajari ini lebih lanjut di materi Network Security).



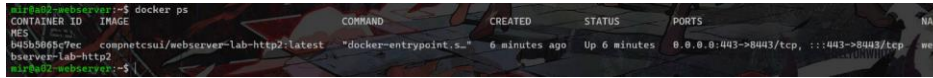
Short - Hello, 1234567890

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Nulla facilisi nullam vehicula ipsum a arcu cursus vitae congue. Neque egestas congue quisque egestas diam in arcu cursus. Vivamus at augue eget arcu. Nibh praesent tristique magna sit amet. Diam vel quam elementum pulvinar etiam non quam. Mauris sit amet massa vitae. Volutpat diam ut venenatis tellus in metus vulputate. Tellus at urna condimentum mattis pellentesque id nibh. Natus et malesuada fames ac turpis egestas integer eget. Viverra suspendisse potenti nullam ac tortor vitae purus faucibus. Augue eget arcu dictum varius dui at consectetur lorem. Sit amet facilisis magna etiam. Ullamcorper a lacus vestibulum sed. Enim facilisis gravida neque convallis.

Step 9: Daftar Docker Container Aktif

Dalam skenario asli, kita perlu memastikan bahwa container kita aktif dan berjalan dengan baik. Docker menyediakan beberapa command yang dapat digunakan untuk memeriksa container yang sedang aktif. Salah satunya adalah 'docker ps', yang digunakan untuk mencetak daftar container.

```
docker ps
```



CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAME
webserver-lab-http2	comnatsul/webserver-lab-http2:latest	docker-entrypoint.s...	6 minutes ago	Up 6 minutes	0.0.0.0:443->8043/tcp, :::443->8043/tcp	webserver-lab-http2

Anda dapat melihat informasi tentang container, termasuk CONTAINER ID dan nama container.

Step 10: Mematikan Docker Container

Best practice dalam menggunakan Docker adalah menghentikan container yang tidak digunakan. Docker container menggunakan resource bahkan saat idle karena kita menjalankan platform sendiri di dalam mesin kita. Meskipun kita bisa saja mematikan seluruh mesin (dalam kasus ini instance GCP) karena mesinnya hanya digunakan untuk menghost web server, kita tidak dapat melakukan hal itu apabila mesinnya ingin digunakan untuk keperluan lain. Anda harus bisa menghentikan container tanpa mematikan seluruh mesin. Command berikut menghentikan Docker container:

```
docker stop <container name>
```

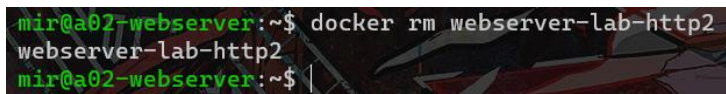


```
mir@a02-webserver:~$ docker stop webserver-lab-http2
webserver-lab-http2
mir@a02-webserver:~$
```

Step 11: Menghapus Docker Container

Meskipun kita sudah menghentikan Docker container sehingga akses ke container pun terputus, file container tersebut masih disimpan oleh Docker. Untuk memastikan container benar-benar dihapus, jalankan command berikut:

```
docker rm <container name>
```



```
mir@a02-webserver:~$ docker rm webserver-lab-http2
webserver-lab-http2
mir@a02-webserver:~$
```

HTTP/2 dan Wireshark

Dalam assignment ini, protokol yang digunakan dalam interaksi antara client dengan web server adalah HTTP/2. Sayangnya, protokol HTTP/2 berjalan di atas protokol TLS. Hal ini menyebabkan message yang dikirim dan diterima oleh client telah terencrypt, sehingga Wireshark tidak dapat membaca isi HTTP/2 message secara langsung. Untuk membuat Wireshark bisa membaca isi dari HTTP/2 message, Anda perlu membuat sebuah pre-master secret key sehingga Wireshark dapat mendecrypt message-messagenya. Untuk membuat pre-master secret key, Anda hanya perlu menambahkan environment variable SSLKEYLOGFILE ke dalam local machine Anda. Nantinya, ketika web browser Anda mengunjungi suatu website yang menggunakan TLS, sebuah log file berisi pre-master secret key yang dispesifikasikan dalam environment variable tersebut akan terbentuk.

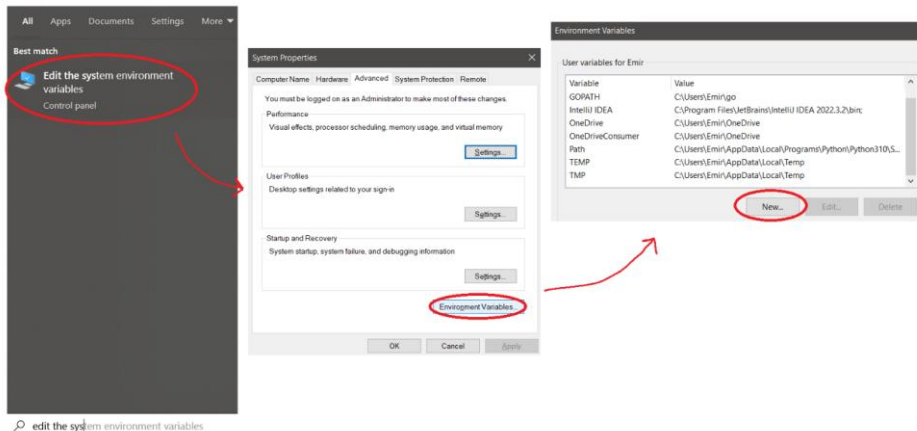
Perhatian: Karena Anda akan mendecrypt seluruh TLS packet yang dikirim dan diterima oleh client, maka **jangan mengakses website-website yang meminta atau menampilkan data sensitif** seperti password, data diri, dsb. Selain itu, **hapuslah environment variable SSLKEYLOGFILE setelah mengerjakan assignment ini**. Namun, **jangan hapus log filenya karena akan dikumpulkan apabila terkena indikasi plagiarisme sebagai bukti pengerjaan**. Karena alasan keamanan tersebut, kami merekomendasikan (tidak wajib) Anda untuk menggunakan **PC lab** dalam mengerjakan assignment ini. Kami tidak akan menanggung resiko yang muncul akibat kelalaian Anda.

Step 1: Membuat Pre-Master Secret Key

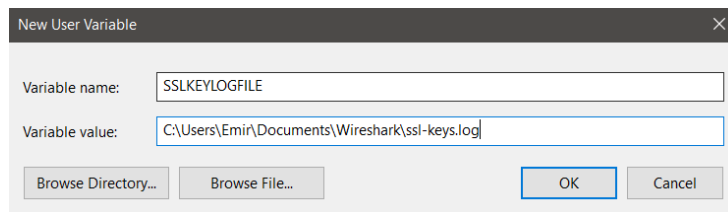
Untuk membuat pre-master secret key, kita perlu menambahkan environment variable bernama SSLKEYLOGFILE dengan value lokasi dari log file. Log file tersebut akan terbentuk secara otomatis setelah Anda mengunjungi suatu website yang menerapkan TLS.

Khusus Pengguna Windows

Khusus pengguna Windows, carilah opsi **“Edit the system environment variables”** pada search bar. Setelah menekan opsi tersebut, akan muncul sebuah window. Tekanlah tombol **“Environment Variables”** dan buatlah user variable baru dengan menekan tombol **“New...”**.



Isi field variable name dengan "SSLKEYLOGFILE" dan variable value dengan lokasi dari log file. Sebagai contoh, Anda bisa memasukkan "C:\Users\<UsernamePC>\Documents\Wireshark\ssl-keys.log". Selanjutnya, tekan "OK" dan tutup seluruh window.



Khusus Pengguna Linux atau Mac

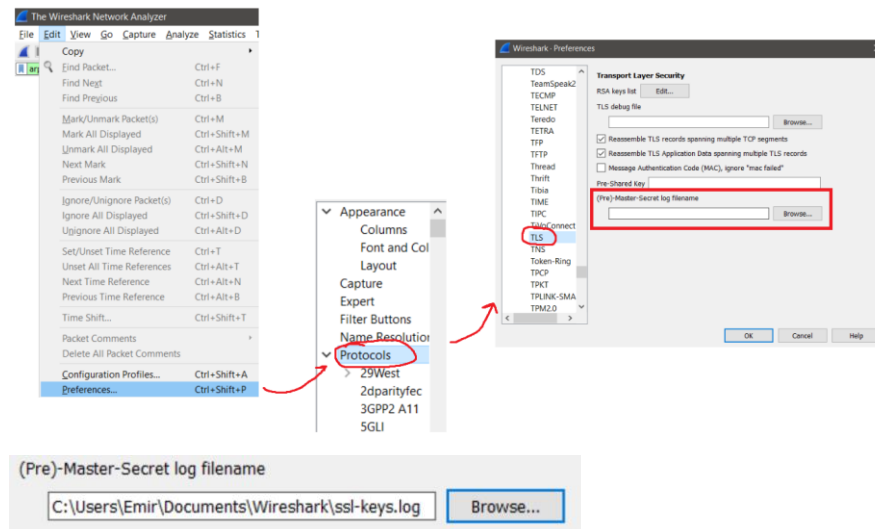
Khusus pengguna Linux, jalankan command `nano ~/.bashrc`. Khusus pengguna MacOS, jalankan command `nano ~/.zshrc`. Tulislah `export SSLKEYLOGFILE=<Lokasi Log File>` dalam line terakhir dari file yang telah Anda buka menggunakan nano. Sebagai contoh, anda bisa menggunakan `~/ssl-key.log` sebagai lokasi dari log file. Setelah menambahkan environment variable, tutup dan buka kembali terminal. Jalankan `echo $SSLKEYLOGFILE` untuk memastikan bahwa environment variable tersebut sudah berhasil ditambahkan.

Memastikan Pre-Master Secret Key Sudah Dibuat

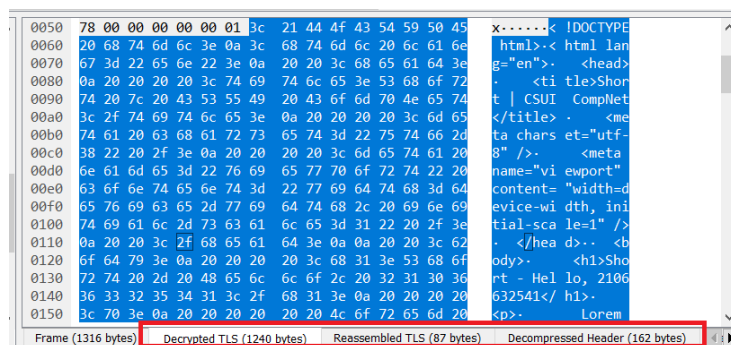
Cobalah untuk mengakses sebuah website dengan protocol HTTP/2, seperti <https://<external-ip-gcp>/<NPM>/short>. Setelah itu, cek di lokasi yang Anda daftarkan sebagai value dari environment variable SSLKEYLOGFILE apakah file lognya sudah terbuat. Apabila file belum ada, cobalah untuk *restart* atau mengganti web browser Anda. **Direkomendasikan** menggunakan **Firefox**.

Step 2: Konfigurasi Wireshark

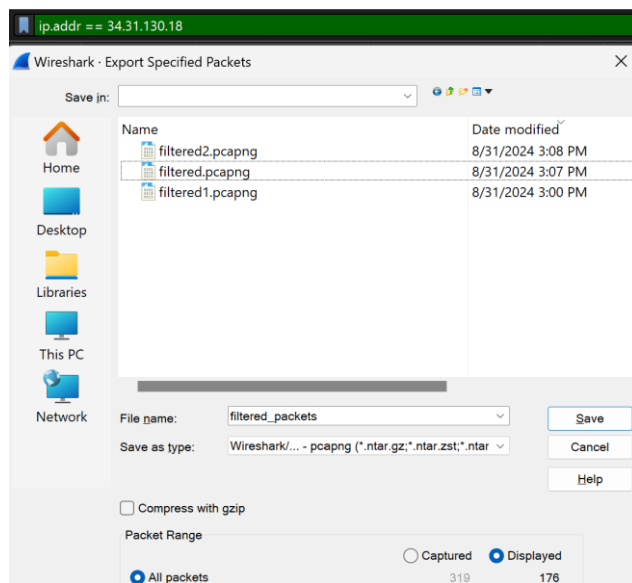
Pada saat ini, log file yang dapat digunakan untuk mendecrypt HTTP/2 message sudah terbentuk. Namun, Wireshark belum mengetahui keberadaan file tersebut. Oleh karena itu, kita perlu memasukan lokasi dari log file ke dalam Wireshark TLS preferences. Wireshark TLS preferences dapat diakses dengan menekan tombol “Edit -> Preferences”, lalu carilah menu “TLS” di bawah menu “Protocol”. Setelah menu TLS terbuka, isilah field “(Pre)-Master-Secret log filename” dengan lokasi log file Anda.



Sekarang, setiap HTTP/2 message yang dikirim dan diterima oleh web browser Anda akan didecrypt secara otomatis oleh Wireshark menggunakan pre-master secret key yang telah Anda buat dengan syarat bahwa capture session dari Wireshark dijalankan sebelum browser dibuka. Hasil decryption (serta decompression, dsb.) dapat diakses di tab-tab selain “Frame” pada suatu HTTP/2 message.



Untuk memperkecil ukuran file Wireshark, Anda dapat mem-filter berdasarkan ip.address eksternal Instance Anda dan kemudian mengekspor file tersebut dengan opsi File > Export Specified Packets dan centang Displayed.



Notes

Selamat! Anda baru saja melakukan deployment dan teardown pada web server menggunakan Docker Engine dalam instance GCP serta membuat key untuk mendecrypt TLS. Kami berharap langkah-langkah di atas dapat membantu Anda untuk melakukan deployment pada web server dan membaca packet-packetnya secara mandiri.

Karena bagian selanjutnya dari assignment ini mengharuskan Anda memiliki web server yang telah dideploy, Anda perlu mengulangi langkah-langkah di atas untuk mendeploy web server yang nantinya dapat diakses dari komputer lokal Anda. Anda tidak perlu mengirimkan apa pun terkait pengerjaan langkah-langkah di atas, tetapi Anda perlu memastikan bahwa web server berfungsi dengan baik untuk dapat melakukan bagian selanjutnya dari assignment ini.

Berikut adalah beberapa hal yang perlu diperhatikan tentang deployment dari web server:

1. Dengan asumsi bahwa Anda menggunakan instance GCP yang sama, Anda tidak perlu menginstal ulang Docker setiap kali ingin men-deploy aplikasi baru. Anda hanya perlu melakukan pull pada image dan menjalankan container.
2. Docker image, seperti git repository, dapat diupdate. Karena kita memerlukan image versi terbaru, Anda harus menarik ulang image sebelum mendeploy container.

3. Docker container akan mati apabila mesin mati. Option '-restart' pada 'docker run' menentukan tindakan apa yang akan dilakukan oleh Docker jika instance GCP Anda direstart atau Docker Engine mengalami error. Dalam option yang digunakan di atas, container akan dijalankan secara otomatis jika terjadi crash atau instance direstart.
4. Jika Anda ingin agar Docker container dapat diakses dari instance GCP, pastikan semua aturan port forwarding telah dibuat. Dalam kasus ini, port forwarding 443:8443 sudah cukup.

Spesifikasi

Catatan: Bagian tugas ini akan diberi penilaian. Semua jawaban **harus dituliskan pada template jawaban yang diberikan** (ditautkan dalam informasi umum). Semua jawaban harus didukung oleh screenshots yang relevan. Anda harus **memberikan highlight pada screenshot** atau **menjelaskan bagian mana dari screenshots (fields, value, dll.)** secara spesifik untuk mendukung jawaban Anda. Jawaban dengan tangkapan layar yang tidak memiliki highlight/deskripsi **tidak akan dinilai**.

Dalam tugas ini, kita akan belajar tentang sebuah application layer yang sering ditemukan: Hypertext Transport Protocol (HTTP). HTTP adalah protokol untuk mendapatkan resources seperti HTML document. HTTP adalah dasar dari data exchange di web dan merupakan protokol client-server, yang berarti request diinisiasi oleh client, biasanya web browser. Document yang lengkap direkonstruksi dari berbagai subdocument yang didapatkan, seperti text, layout descriptions, image, video, script, dan masih banyak lagi. Dengan memeriksa beberapa HTTP message, kami harap Anda dapat memahami protokol ini dan bagaimana protokol itu membantu eksplorasi web.

Sebelum memulai assignment ini, lakukan hal berikut:

1. Screenshot IP address dari komputer lokal Anda (hanya akan valid di jaringan lokal Anda). Hal ini dapat dilakukan dengan menjalankan command berikut (pastikan Anda mengambil screenshot dari interface yang akan digunakan di Wireshark)

```
# Windows  
ipconfig  
# macOS  
ipconfig  
# Linux  
ip -a
```

2. Ambil screenshot alamat IP Eksternal instance GCP dari dashboard GCP.
3. Pastikan web server Anda aktif dan berjalan di instance GCP Anda.

Steps

Pada bagian ini, Anda akan diminta untuk melakukan capture menggunakan Wireshark pada HTTP/2 request dan response dan menyimpannya dalam file pcapng. Anda tidak perlu mengumpulkan key log file-nya, tetapi Anda perlu menyimpan key log filenya.

Tips: Penggunaan browser Firefox direkomendasikan, namun tidak diwajibkan.

Basic HTTP Analysis

1. Siapkan tab di web browser Anda di komputer lokal.
2. Buka URL berikut (ubah placeholder dengan informasi yang relevan).

```
https://<instance-external-ip-address>/<NPM>/short  
# Example for IP 34.56.78.90 and NPM 2006596062  
# https://34.56.78.90/2006596062/short
```

3. Jika server memberikan HTML response, lanjutkan ke langkah berikutnya. Jika tidak, periksa kembali koneksi atau konfigurasi web server/instance GCP Anda.
4. Mulai Wireshark dan mulai capture session. Tunggu setidaknya 20 detik.
5. Hard refresh halaman web sebelumnya. Pada browser Chromium atau Firefox, tekan **Ctrl** (**⌘Command for macOS**) + **Shift + R**. Pada safari, tekan **⌘Command + Opt + R** (akan dikonfirmasi).
6. Tunggu setidaknya 20 detik untuk memastikan bahwa paket dari halaman web yang direfresh sudah tercapture dengan benar.
7. Hentikan Wireshark capture dan simpan hasilnya dengan nama bebas (berkas ini tidak dikumpulkan dalam kondisi saat ini). **Simpan berkas ini baik-baik sebagai cadangan apabila Anda salah melakukan filtrasi.**
8. Terapkan filter dalam Wireshark capture sehingga hanya koneksi ke/dari external IP address milik instance GCP yang ditampilkan.
9. Buka menu **File > Export Specified Packets** untuk mengekspor berkas yang telah difiltrasi dan siap dikumpulkan.
10. Pastikan **radio button Displayed** dipilih agar Wireshark hanya memasukkan paket yang telah difiltrasi dan ditampilkan kepada Anda.
11. Simpan berkas yang sudah difiltrasi ini dengan nama sebagai berikut:

A02_[NPM]_BasicHTTPAnalysis.pcapng

HTTP Content Typing

1. Siapkan tab di web browser Anda di komputer lokal.
2. Di tab pertama, buka halaman yang digunakan di bagian "Basic HTTP Analysis" (<https://<instance-external-ip-address>/<NPM>/short>).
3. Di tab kedua, buka URL ini:

```
https://<instance-external-ip-address>/<NPM>/json  
# Example for IP 34.56.78.90 and NPM 2006596062  
# https://34.56.78.90/2006596062/json
```

4. Mulai Wireshark dan mulai capture session. Tunggu setidaknya 20 detik.
5. Hard refresh kedua halaman web sebelumnya. Pada browser Chromium atau Firefox, tekan **Ctrl** (**⌘Command for macOS**) + **Shift + R**. Pada safari, tekan **⌘Command + Opt + R** (akan dikonfirmasi).
6. Tunggu setidaknya 20 detik untuk memastikan bahwa paket dari halaman web yang direfresh sudah tercapture dengan benar.

7. Hentikan Wireshark capture dan simpan hasilnya dengan nama bebas (berkas ini tidak dikumpulkan dalam kondisi saat ini). **Simpan berkas ini baik-baik sebagai cadangan apabila Anda salah melakukan filtrasi.**
8. Terapkan filter dalam Wireshark capture sehingga hanya koneksi ke/dari external IP address milik instance GCP yang ditampilkan.
9. Buka menu **File > Export Specified Packets** untuk mengekspor berkas yang telah difiltrasi dan siap dikumpulkan.
10. Pastikan *radio button* **Displayed** dipilih agar Wireshark hanya memasukkan paket yang telah difiltrasi dan ditampilkan kepada Anda.
11. Simpan berkas yang sudah difiltrasi ini dengan nama sebagai berikut:

A02_[NPM]_HTTPContentTyping.pcapng

Retrieving Long Documents

1. Siapkan tab di web browser Anda di komputer lokal.
2. Buka URL berikut:

```
https://<instance-external-ip-address>/<NPM>/long  
# Example for IP 34.56.78.90 and NPM 2006596062  
# https://34.56.78.90/2006596062/long
```

1. Jika server memberikan HTML response, lanjutkan ke langkah berikutnya. Jika tidak, periksa kembali koneksi atau konfigurasi web server/instance GCP Anda.
3. Mulai Wireshark dan mulai capture session. Tunggu setidaknya 20 detik.
4. Hard refresh halaman web sebelumnya. Pada browser Chromium atau Firefox, tekan **Ctrl** (**⌘Command for macOS**) + **Shift + R**. Pada safari, tekan **⌘Command + Opt + R** (akan dikonfirmasi).
5. Tunggu setidaknya 20 detik untuk memastikan bahwa paket dari halaman web yang direfresh sudah tercapture dengan benar.
6. Hentikan Wireshark capture dan simpan hasilnya dengan nama bebas (berkas ini tidak dikumpulkan dalam kondisi saat ini). **Simpan berkas ini baik-baik sebagai cadangan apabila Anda salah melakukan filtrasi.**
7. Buka menu **File > Export Specified Packets** untuk mengekspor berkas yang telah difiltrasi dan siap dikumpulkan.
8. Pastikan *radio button* **Displayed** dipilih agar Wireshark hanya memasukkan paket yang telah difiltrasi dan ditampilkan kepada Anda.
9. Hentikan Wireshark capture dan simpan hasilnya sebagai

A02_[NPM]_RetrievingLongDocuments.pcapng

10. Terapkan filter dalam Wireshark capture sehingga hanya koneksi ke/dari external IP address milik instance GCP yang ditampilkan.

Content with Embedded Objects

1. Siapkan tab di web browser Anda di komputer lokal.
2. Buka URL berikut:

```
https://<instance-external-ip-address>/<NPM>/embedded-contents  
# Example for IP 34.56.78.90 and NPM 2006596062  
# https://34.56.78.90/2006596062/embedded-contents
```

3. Jika server memberikan HTML response, lanjutkan ke langkah berikutnya. Jika tidak, periksa kembali koneksi atau konfigurasi web server/instance GCP Anda.
4. Mulai Wireshark dan mulai capture session. Tunggu setidaknya 20 detik.
5. Hard refresh halaman web sebelumnya. Pada browser Chromium atau Firefox, tekan **Ctrl** (**⌘Command for macOS**) + **Shift** + **R**. Pada safari, tekan **⌘Command** + **Opt** + **R** (akan dikonfirmasi).
6. Tunggu setidaknya 20 detik untuk memastikan bahwa paket dari halaman web yang direfresh sudah tercapture dengan benar.
7. Hentikan Wireshark capture dan simpan hasilnya dengan nama bebas (berkas ini tidak dikumpulkan dalam kondisi saat ini). **Simpan berkas ini baik-baik sebagai cadangan apabila Anda salah melakukan filtrasi.**
8. Terapkan filter dalam Wireshark capture sehingga hanya koneksi ke/dari external IP address milik instance GCP yang ditampilkan.
9. Buka menu **File > Export Specified Packets** untuk mengekspor berkas yang telah difiltrasi dan siap dikumpulkan.
10. Pastikan **radio button Displayed** dipilih agar Wireshark hanya memasukkan paket yang telah difiltrasi dan ditampilkan kepada Anda.
11. Simpan berkas yang sudah difiltrasi ini dengan nama sebagai berikut:

A02_[NPM]_EmbeddedObjects.pcapng

Persistent HTTP (Caching)

1. Siapkan tab di web browser Anda di komputer lokal.
2. Buka halaman yang digunakan di bagian "Basic HTTP Analysis" (<https://<instance-external-ip-address>/<NPM>/short>).
3. Jika server memberikan HTML response, lanjutkan ke langkah berikutnya. Jika tidak, periksa kembali koneksi atau konfigurasi web server/instance GCP Anda.
4. Mulai Wireshark dan mulai capture session. Tunggu setidaknya 20 detik.
5. Hard refresh halaman web sebelumnya. Pada browser Chromium atau Firefox, tekan **Ctrl** (**⌘Command for macOS**) + **Shift** + **R**. Pada safari, tekan **⌘Command** + **Opt** + **R** (akan dikonfirmasi).
6. Refresh halaman sekali lagi. Kali ini, jangan gunakan hard refresh. Tombol refresh (**f5**) seharusnya cukup untuk langkah ini.
7. Tunggu setidaknya 20 detik untuk memastikan bahwa paket dari halaman web yang direfresh sudah tercapture dengan benar.
8. Hentikan Wireshark capture dan simpan hasilnya dengan nama bebas (berkas ini tidak dikumpulkan dalam kondisi saat ini). **Simpan berkas ini baik-baik sebagai cadangan apabila Anda salah melakukan filtrasi.**

9. Terapkan filter dalam Wireshark capture sehingga hanya koneksi ke/dari external IP address milik instance GCP yang ditampilkan.
10. Buka menu **File > Export Specified Packets** untuk mengekspor berkas yang telah difiltrasi dan siap dikumpulkan.
11. Pastikan *radio button* **Displayed** dipilih agar Wireshark hanya memasukkan paket yang telah difiltrasi dan ditampilkan kepada Anda.
12. Simpan berkas yang sudah difiltrasi ini dengan nama sebagai berikut:

A02_[NPM]_Caching.pcapng

[37 Points] Basic HTTP Analysis

1. [7] Temukan frame yang berisi HTTP request. Machine mana yang bertindak sebagai source dari pesan ini (apakah itu local machine atau web server Anda)? Jelaskan bagaimana Anda dapat menyimpulkannya dari informasi yang tersedia.
2. [6] Identifikasi dan laporkan semua komponen pesan yang dikembalikan oleh server! Apakah seluruh pesan yang dikembalikan oleh server dimuat dalam satu paket?
3. [6] Apa jenis HTTP request yang digunakan? Apakah request ini dianggap safe (tidak ada side effect)? Jelaskan alasannya!
4. [6] Bagaimana cara klien mengetahui fungsi dari bagian tertentu dalam pesan yang dikembalikan (misal sebagai header atau sebagai data)? Tandai bagian yang menunjukkan informasi tersebut!
5. [6] Apa status code dari HTTP response dalam packet yang ditemukan? Apa arti dari status code dan phrase itu? Tandai bagian yang menunjukkan informasi tersebut!
6. [6] Berapa body size dari HTTP response ke request /short?

[14 Points] HTTP Content Typing

1. [7] Bandingkan HTTP response dari dua request yang berbeda. Anda mungkin memperhatikan bahwa browser Anda dapat merender response secara berbeda. Bagaimana browser Anda mengetahui tentang content type yang diterimanya? (Petunjuk: periksa HTTP response).
2. [7] Apa content type dari URL /short? Bagaimana dengan content type dari URL /json?

[14 Points] Retrieving Long Documents

1. [7] Dapatkan seluruh payload dari HTTP response message muat dalam satu frame? Jika ya, frame mana (sebutkan frame number) yang berisi keseluruhan response? Jika tidak, sebutkan semua frame number yang berpartisipasi dalam pengiriman HTTP response.
2. [7] Frame mana yang berisi HTTP response headers? (Anda mungkin perlu memeriksa contents dari frame).

[14 Points] Content with Embedded Objects

1. [7] Berapa banyak HTTP request yang dibuat dari source? Apa kegunaan setiap request (konten apa yang diminta dari setiap request)?
2. [7] Perhatikan konten yang Anda unduh tersebut! Bisakah Anda menentukan apakah seluruh konten tersebut diambil secara bersamaan (secara paralel) atau serial, atau ada sebagian yang diunduh hanya setelah proses pengunduhan lainnya selesai? Jelaskan berdasarkan

Commented [AZ1]: Yang dilabel aman dari gpt, pertanyaan sudah di copas ke chatgpt dan ngga langsung dapet jawabannya

Commented [AZ2R1]: biru = tambahan

bukti yang didapatkan! [Petunjuk: Anda dapat membuktikannya dengan membuat grafik trafik berdasarkan data dari Wireshark, di mana sumbu Y menggambarkan objek request dan sumbu X menunjukkan waktu.]

[21 Points] Persistent HTTP (Caching)

1. [5] Apa status code dan phrase dari HTTP response kedua? Apakah payload dikirim dari server dalam HTTP response tersebut? Jelaskan!
2. [6] Ada beberapa cara client dan server dapat menyepakati proses caching. Field apa dalam HTTP request kedua yang akan dicek untuk memutuskan apakah server harus mengirim payload?
3. [5] Dari mana value dari field HTTP request kedua yang telah Anda identifikasi di nomor 2 berasal? Secara spesifik, field mana yang menjadi acuan untuk mengisi field tersebut? Tandai bagian yang menunjukkan informasi tersebut!.
4. [5] Mengapa force reload dapat memaksa server untuk mengembalikan response? Apa yang membedakan HTTP request pertama dan kedua sehingga request pertama memiliki payload sedangkan yang kedua tidak? Jelaskan dan tandai bagian yang menunjukkan informasi tersebut!.

Informasi Terkait Pengumpulan Berkas

Anda hanya perlu mengirimkan bagian "Spesifikasi" dari tugas ini. Ini adalah file yang perlu diserahkan:

1. Lima file Wireshark (.pcapng) yang sudah dilakukan filtrasi sebagai berikut:
 - a. **A02_[NPM]_BasicHTTPAnalysis.pcapng**
 - b. **A02_[NPM]_HTTPContentType.pcapng**
 - c. **A02_[NPM]_RetrievingLongDocuments.pcapng**
 - d. **A02_[NPM]_EmbeddedObjects.pcapng**
 - e. **A02_[NPM]_Caching.pcapng**
2. Laporan (.pdf) yang berisi semua jawaban dan tangkapan layar untuk setiap pertanyaan di "Spesifikasi". Ubah nama file sebagai **A02_[NPM].pdf**.

Peraturan

Keterlambatan

Anda diharapkan dapat mengumpulkan hasil pekerjaan yang dilakukan sebelum batas waktu pengumpulan. Jika terdapat kondisi di mana Anda terpaksa terlambat mengumpulkan hasil pekerjaan, terdapat jangka waktu tambahan di mana Anda masih diperbolehkan mengumpulkan hasil pekerjaan dengan konsekuensi tertentu. Jika X adalah durasi setelah batas waktu pengumpulan yang ditetapkan sampai waktu Anda mengumpulkan hasil pekerjaan, Anda akan menerima penalti nilai pekerjaan sebagaimana diatur pada peraturan berikut ini:

- | | |
|---|--|
| • $X < 10$ menit | : Tidak ada penalti |
| • $10 \text{ menit} \leq X < 2 \text{ jam}$ | : 25% penalti |
| • $2 \text{ jam} \leq X < 4 \text{ jam}$ | : 50% penalti |
| • $4 \text{ jam} \leq X < 6 \text{ jam}$ | : 75% penalti |
| • $X \geq 6 \text{ jam}$ | : Cut-off (Pekerjaan anda tidak akan diterima) |

Plagiarisme

Anda diperbolehkan berdiskusi tentang pekerjaan Anda dengan peserta kuliah lain atau pihak lainnya, namun Anda harus memastikan bahwa **semua pekerjaan yang dikumpulkan adalah murni hasil pekerjaan Anda sendiri**. Anda dilarang keras melakukan tindak plagiarisme atau kecurangan akademik lainnya. Menurut kamus daring Merriam-Webster, plagiarisme berarti:

- Mencuri dan mengklaim (ide atau kata orang lain) sebagai milik sendiri
- Menggunakan hasil (karya/pekerjaan orang lain) sebagai milik sendiri
- Melakukan pencurian literatur/sastra
- Merepresentasikan ulang sebuah ide/produk yang sudah ada sebagai sesuatu yang bersifat baru dan orisinal.

Tim pengajar memiliki hak untuk meminta klarifikasi terkait dugaan ketidakjujuran akademik, terutama plagiarisme, dan memberikan konsekuensi berupa **pengurangan nilai hasil pekerjaan atau pencabutan nilai (nilai diubah menjadi nol) untuk hasil pekerjaan yang terkonfirmasi dikerjakan secara tidak jujur**.