PROPOSAL PRA SKRIPSI
IMPLEMENTASI MITMPROXY UNTUK HTTPS FILTERING

Oleh:
Aldzikri Dwijayanto Prathama
195410189

PROGRAM STUDI INFORMATIKA
PROGRAM SARJANA
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS TEKNOLOGI DIGITAL INDONESIA
YOGYAKARTA
2022

Daftar Isi

Lembar Persetujuan

Judul	:	Penerapan Mitmproxy Untuk HTTPS Filtering
Nama	:	Aldzikri Dwijayanto Prathama
NIM	:	195410189
Program Studi	:	Informatika
Program	:	Sarjana
Semester	:	VI(Enam)

Telah memenuhi syarat dan disetujui untuk diseminarkan dihadapan dosen penguji seminar PROPOSAL PRA SKRIPSI.

Yogyakarta, 2022

Dosen Pembimbing,

Agung Budi Prasetyo S.Kom., M.Kom.

NIDN : 0003087106

1 Latar Belakang

Hampir seluruh perangkat saat ini terhubung dengan internet, seperti *smartphone*, komputer, dan *smart TV*. Meskipun tekonologi internet sangatlah bermanfaat, tetapi teknologi ini juga menimbulkan ancaman keamanan.

Dari survey yang dilakukan, mayoritas developer tidak ragu melakukan *personalized-ads*. Jenis iklan ini menggunakan data pengguna untuk memilihkan iklan yang dianggap relevan berdasarkan data pengguna. Hal ini tidak mengherankan karena jenis iklan ini menawarkan pendapatan yang lebih tinggi dibandingkan iklan biasa, karena peluang pengguna membuka iklan yang ditawarkan lebih besar(Tahaei, Frik dan Vaniea 2021). Penelitian lain juga menunujukkan bahwa 77% aplikasi Android gratis menggunakan *ad library*(He et al. 2018; Jin et al. 2021).

Iklan-iklan yang ditampilkan sering kali sangat intrusif dan menggangu. Contohnya seperti iklan yang menutupi konten, iklan melalui *pop-up*, atau iklan yang otomatis mengarahkan pengunjung ke *tab* baru yang memuat iklan. Atau bahkan menampilkan iklan yang menjurus ke arah konten dewasa.

Masalah lain selain iklan yang mengganggu, adalah masalah keamanan. Banyak layanan yang memasang iklan juga menggunakan layanan *web analytics* untuk melakukan *tracking* terhadap pengunjung atau penggunanya. Layanan web analytics inilah yang dapat menjadi celah keamanan. Contohnya pada tahun 2019, layanan Google Analytics digunakan untuk mengiklankan website phising(Osborne 2019). Sedangkan di tahun 2020, peretas memasukkan kode berbahaya ke dalam website yang diretas, yang mana kode tersebut mengumpulkan informasi kredit pengguna dan mengirimkannya menggunakan Google Analytics, kemudian peretas akan mengakses data kredit yang dikumpulkan di akun Google Analytics miliknya(Lakshmanan 2020).

Hal-hal tersebut bisa dicegah dengan menggunakan metode DNS *ad-blocking*, dengan metode ini, kita bisa melakukan pemblokiran domain yang melakukan *tracking*, domain yang digunakan sebagai tempat menyimpan kode JavaScript yang berbahaya, atau domain milik penyedia iklan. Tetapi metode ini hanya bisa melakukan pemblokiran berdasarkan nama domain, sehingga jika iklan atau kode JavaScript berada pada domain yang sama dengan website utama, maka kita harus mengizinkan iklan, dan kode JavaScript berjalan, atau memblokir domain website tersebut sehingga tidak bisa diakses.

Metode lain adalah menggunakan *add-on* peramban, seperti *UBlock Origin*. Metode ini mampu melakukan *blocking* terhadap iklan atau kode JavaScript yang berbahaya, walaupun keduanya berada pada domain yang sama dengan website utama. Sayangnya *add-on* ini hanya bisa di-*install* pada peramban tertentu, seperti Mozilla Firefox, dan Chrome Desktop.

Berdasarkan permasalahan yang disebutkan di atas, tujuan saya adalah membangun sistem HTTPS filtering dengan menggunakan Mitmproxy. Sistem ini bertujuan untuk menyaring konten yang tidak diinginkan agar tidak diakses.

Filtering dengan menggunakan Mitmproxy saya pilih karena tools ini merupakan proxy, sehingga bisa digunakan di banyak perangkat seperi metode DNS *ad-blocking*, tetapi bisa di buat sedemikian rupa dengan menuliskan *script* Python sehingga mampu melakukan filtering seperti *add-on* pada browser, atau bahkan melebihinya.

Selain itu kelebihan Mitmproxy yang lain adalah software ini bersifat *open source* dan gratis. Software ini bisa digunakan sebagai alat untuk mencegat, inspeksi, dan memodifikasi lalu lintas web, seperti HTTP/1, HTTP/2, WebSockets, atau protokol lainnya yang dilindungi oleh SSL/TLS.

2 Rumusan Masalah

Berdasarkan latar belakang yang diuraikan di atas, maka dapat dirumuskan beberapa pokok permasalahan sebagai berikut:

1. Bagaimana mengimplementasikan MITM-Proxy sebagai *ad-blocker*?
2. Bagaimana mengkonfigurasi perangkat untuk menggunakan proxy MITM-Proxy?
3. Apakah MITM-Proxy lebih efektif daripada *add-on ad-block* di browser dan DNS *ad-blocking*?

3 Ruang Lingkup

Untuk membatasi cakupan dari sistem HTTPS filtering dengan MITM-Proxy ini maka dibuat ruang lingkup sebagai berikut:

1. Sistem yang dibuat merupakan sistem proxy yang bersifat non-transparent.
2. Sistem digunakan untuk menyaring *response* yang melalui MITM-Proxy.
3. Sistem bertujuan untuk menghilangkan konten-konten yang tidak diinginkan sehingga konten tersebut tidak dimuat dan/atau dijalankan oleh perangkat *client*.
4. Script yang akan dimuat oleh MITM-Proxy ditulis dengan menggunakan bahasa pemrograman Python.
5. Server proxy yang digunakan adalah MITM-Proxy.
6. Sistem akan diujikan dengan menggunakan halaman web *dummy*.

4 Tujuan Penelitian

Tujuan yang ingin dicapai dari penelitian ini adalah mengimplementasikan MITM-Proxy sebagai server proxy non-transparan. Serta membandingkannya dengan metode *ad-blocking* lain.

Sistem ini dirancang sebagai filter konten yang berbentuk proxy, sehingga bisa digunakan di perangkat manapun selama perangkat tersebut memiliki setelan proxy.

Script yang digunakan sebagai *ad-blocking* yang akan digunakan di server proxy MITM-Proxy ditulis dengan menggunakan bahasa pemrograman Python. Sedangkan Sistem Operasi yang digunakan untuk menjalankan server adalah Sistem Operasi berbasis Linux.

5 Dasar Teori

1. Mitmproxy

Mitmproxy adalah seperangkat alat yang menyediakan proxy yang mampu mencegat SSL/TLS untuk protokol HTTP/1, HTTP/2, dan WebSockets(Cortesi et al. 2010–).

Fitur dari Mitmproxy antara lain mencegat *request* dan *response* dari HTTP dan HTTPS dan memodifikasinya, Kemampuan untuk mengubah lalu lintas HTTP dengan menggunakan *script* Python, dll(Cortesi et al. 2010–).

2. HTTPS

HTTPS (Hypertext Transfer Protocol Secure) adalah protokol komunikasi internet yang melindungi integritas dan kerahasiaan data antara komputer pengguna dan situs. Pengguna menginginkan pengalaman online yang aman dan bersifat pribadi saat menggunakan situs.

Data yang dikirim menggunakan HTTPS diamankan melalui protokol Transport Layer Security (TLS), yang memberikan tiga lapis perlindungan utama:

(a) Enkripsi

Mengenkripsi data pertukaran untuk menjaga keamanannya dari penyadap. Artinya, saat pengguna menjelajahi situs, tidak ada yang dapat ”menguping” percakapan, melacak aktivitas di berbagai halaman, atau mencuri informasi mereka.

(b) Integritas data

Data tidak dapat diubah atau dirusak selama transfer, dengan sengaja atau tidak, tanpa terdeteksi.

(c) Autentikasi

Membuktikan bahwa pengguna berkomunikasi dengan situs yang diinginkan. Protokol tersebut melindungi dari serangan *man in the middle* dan membangun kepercayaan pengguna, yang dapat memberikan keuntungan lain untuk bisnis.

(Google Developer 2022)

3. Web Content Filtering

Web content filtering merupakan saringan konten website yang digunakan oleh perorangan, kelompok, maupun organisasi untuk melakukan penyaringan terhadap situs-situs yang tidak diperbolehkan oleh pihak berwenang maupun yang tidak berhubungan dengan tujuan bisnis atau organisasi agar tidak dapat diakses(Dewi dan Islami 2021).

4. Proxy Server

Proxy server (peladen proxy) adalah sebuah komputer server atau program komputer yang dapat bertindak sebagai komputer lainnya untuk melakukan request terhadap content dari Internet atau Intranet(Towidjojo 2013).

Proxy server bertindak sebagai gateway terhadap dunia ini Internet untuk setiap komputer clien. Proxy server tidak terlihat oleh komputer client: seorang pengguna yang berinteraksi dengan Internet melalui sebuah proxy server tidak akan mengetahui bahwa sebuah proxy server sedang menangani request yang dilakukannya. Web server yang menerima request dari proxy server akan menginterpretasikan request-request tersebut seolah-olah request itu datang secara langsung dari komputer client, bukan dari proxy server(Towidjojo 2013).

6 Tinjauan Pustaka

Dalam mengimplementasikan MITM-Proxy untuk HTTPS *filtering*, sebagai pedoman dan pembanding maka digunakan beberapa pustaka yang berkaitan dengan *ad-blocking*. Pustaka yang digunakan sebagai rujukan antara lain:

Dari penelitian yang dilakukan oleh Uni Dewi Samima 2021, yang berjudul 'Perancangan Sistem Blocking Situs Berbahaya Menggunakan Pi-Hole Berbasis Docker'. Tujuan dari dilakukannya penelitian ini adalah untuk mengetahui rancangan sistem *blocking* situs berbahaya dengan menggunakan *pi-hole* berbasis docker dan *openvpn*, dan menguji kemampuan *blocking* situs dan performa dari *pihole* dalam *docker container*.

Yusoff dan Baharudin 2020, Melakukan penelitian yang berjudul *'Virtual Private Network Server and Adblock Server using Raspberry Pi with Parental Control'*. Penelitian tersebut bertujuan untuk membangun server openvpn menggunakan Raspberry-pi dan menggunakan pi-hole sebagai sistem *ad-block*. Selain itu penelitian ini juga memanfaatkan kemampuan pi-hole memahami *regular expression* sebagai *parental control* untuk mencegah diaksesnya situs-situs dewasa.

Prosiding dengan judul *'Low-cost Security Solution for Micro, Small and Medium Enterprises'*yang ditulis oleh S, A dan P 2020. Tujuan dari penelitian tersebut adalah membuat sistem keamanan jaringan yang murah untuk bisnis kelas menengah ke bawah, dengan menggunakan Raspberry-pi sebagai perangkat kerasnya. Sedangkan di bagian perangkat lunak, digunakan Dnsmasq. Dnsmasq digunakan sebagai DNS *filtering* dari domain yang dianggap berbahaya.

Prosiding oleh Wahyudi, Diansyah dan Handoko 2020, dengan judul 'PEMANFAATAN PI-HOLE DALAM MELAKUKAN BLOK IKLAN PADA WEBSITE DI SMK TIK DARUSSALAM'. Penelitian ini membahas bagaimana cara mengimplementasikan pi-hole di Ubuntu server sebagai DNS *blocking* pada jaringan SMK TIK Darussalam Medan.

Dari penelitian yang dilakukan oleh Habibi 2022 dengan judul 'OPTIMALISASI INTERNET WARGA MENGGUNAKAN KOMBINASI TYPE ANTRIAN DAN SISTEM PIHOLE'. Penelitian ini membahas tentang pemasangan infrastruktur jaringan internet dengan menggunakan *Point to Multipoint*, meminimalkan hilangnya paket data pada saat transmisi menggunakan RED dan PCQ dan menggunakan Sistem pi-hole di server untuk meminimalkan iklan yang muncul di protokol UDP.

Tabel 1:

Penulis	**Judul Penelitian**	**Tools**	**Hasil**
Uni Dewi Samima 2021	'Perancangan Sistem Blocking Situs Berbahaya Menggunakan Pi-Hole Berbasis Docker'	pi-hole, OpenVPN, dan docker	Server vpn dengan menggunakan protokol OpenVPN, yang menggunakan pi-hole sebagai sistem blocking domain berbahaya. Server tersebut berjalan di dalam docker container.
Yusoff dan Baharudin 2020	'Virtual Private Network Server and Adblock Server using Raspberry Pi with Parental Control'	pi-hole dan raspberry pi	DNS Filtering dengan menggunakan pi-hole yang berjalan di atas raspberry pi. Selain itu dengan menggunakan regular expression, pi-hole dapat mengenali situs-situs dewasa sehingga dapat menjadi solusi parental control.

Bersambung ke halaman berikutnya

Tabel 1: (Bersambung)

S, A dan P 2020	'Low-cost Security Solution for Micro, Small and Medium Enterprises'	Dns-masq	Server DNS dengan menggunakan Dnsmasq yang berjalan di atas raspberry pi. Sistem ini dapat menjadi pilihan untuk mencegah client mengakses domain yang berbahaya. Sistem ini dapat menjadi pilihan untuk diimplementasikan di usaha kelas menengah ke bawah.
Wahyudi, Diansyah dan Handoko 2020	'PEMANFAATAN PI-HOLE DALAM MELAKUKAN BLOK IKLAN PADA WEBSITE DI SMK TIK DARUSSALAM'	pi-hole, ubuntu server	Server DNS pi-hole yang berjalan di atas sistem operasi ubuntu server. Sistem ini diimplementasikan oleh penulis di jaringan SMK TIK Darussalam Medan.
Habibi 2022	'OPTIMALISASI INTERNET WARGA MENGGUNAKAN KOMBINASI TYPE ANTRIAN DAN SISTEM PIHOLE'	pi-hole	DNS filtering diimplementasikan di jaringan sehingga mengurangi penggunaan data pada jaringan internet warga, sehingga jaringan tersebut lebih optimal.

Penelitian yang sudah dilakukan di atas, memiliki tujuan yang serupa dengan penelitian ini yaitu *ad-blocking*. Namun di penelitian ini, yang membahas implementasi MITM-Proxy untuk https filtering, menggunakan MITM-Proxy sebagai perangkat lunak utamanya.

Perbedaan lainnya dari penelitian yang sudah dilakukan adalah tujuan dari penelitian ini adalah membangun sistem yang tidak hanya menyaring berdasarkan domain, tetapi juga menyaring berdasarkan konten yang diakses. Konten yang dimaksud adalah seperti file HTML, CSS, dan JavaScript yang utamanya digunakan untuk menampilkan web, juga respon dari server seperti JSON yang utamanya digunakan di *API*.

7 Analisis Kebutuhan

Dalam proses pembuatan sistem ini, membutuhkan *software* dan *hardware* yang berperan sebagai *server* proxy, analisis kebutuhan dalam pembuatan sistem ini antara lain:

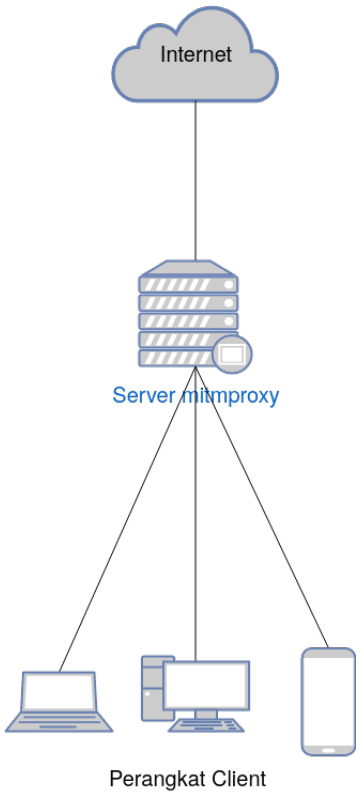
1. Kebutuhan perangkat lunak
    (a) Sistem Operasi: NixOS Linux
    (b) *Proxy Server*: mitmproxy
    (c) Bahasa pemrograman: Python3
    (d) *Text editor*: Neovim
    (e) *Browser*: Mozilla Firefox
2. Kebutuhan perangkat keras
    (a) Komputer *Server*

- CPU: AMD Ryzen 5 3400G
- RAM: 16 GB
- *Storage*: NVME 240GB
- GPU: Radeon Vega Graphics

8 Pemodelan yang digunakan

8.1 Topologi Jaringan

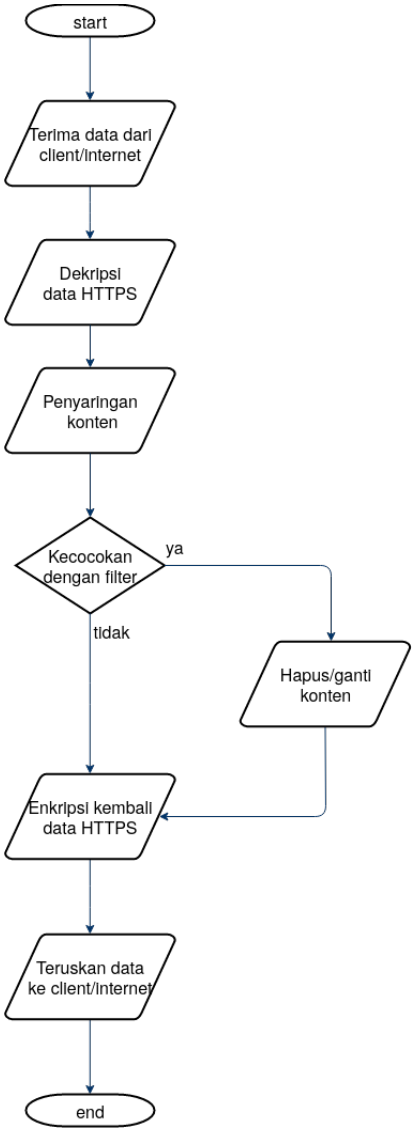
Pemodelan yang digunakan merepresentasikan susunan dari komputer yang ada pada jaringan. Pada perancangan jaringan yang akan diimplementasikan mitmproxy menggunakan pemodelan topologi jaringan.



Gambar 1: Topologi Jaringan Sistem

Sistem yang dibuat merupakan server proxy, sehingga perangkat-perangkat client akan terhubung ke proxy terlebih dahulu sebelum terhubung pada internet.

8.2 Algoritma



Gambar 2: Algoritma Sistem Mitmproxy

Untuk melakukan enkripsi diperlukan *certificate*. *Certificate* dapat diperoleh dari yang sudah dibuat oleh mitmproxy, atau dapat juga dibuat dengan menggunakan

OpenSSL. Untuk membuat *certificate* dengan menggunakan OpenSSL, Gunakan perintah:

```
openssl req -key private_key -x509 -new -days days -out namafile
```

Setelah *certificate* didapatkan, *certificate* tersebut perlu untuk diinstall di perangkat *client*.

Sedangkan untuk melakukan penyaringan konten, memanfaatkan *library* Python BeautifulSoup dan library JSON *built-in* di Python. *Library* BeautifulSoup digunakan untuk mencari komponen-komponen HTML secara idiomatis, yang digunakan untuk menampilkan halaman web. Sedangkan library JSON digunakan untuk mem-*parsing* data JSON, karena ada beberapa halaman web yang memanfaatkan FetchAPI untuk mengunduh kontennya. FetchAPI biasanya berkomunikasi dengan format data JSON. Dengan menggunakan *library* ini diharapkan pembuatan script python untuk memfilter konten yang ditentukan menjadi mudah, karena tidak harus mencari komponen-komponen HTML dengan menuliskan *regular-expression*.

Berikut ini merupakan contoh penggunaan *library* BeautifulSoup.

```
1  from mitmproxy import http
2  from mitmproxy import ctx
3  from bs4 import BeautifulSoup
4
5  class ReplaceHTML:
6    filter = "ads"
7
8    def response(self, flow: http.HTTPFlow):
9      html = BeautifulSoup(flow.response.content, "lxml")
10     for div in html.find_all('div', attrs={"class":filter}):
11       div.decompose()
12     flow.response.content = str(html).encode("utf8")
13
14 addons = [ReplaceHTML()]
```

Kode di atas menggunakan library yang disediakan oleh mitmproxy, dan BeautifulSoup. Di kode tersebut terdapat *events* response. *Events* ini dipicu sebelum response sampai di perangkat *client*. Kemudian *content* dari response di-*parsing* oleh BeautifulSoup menggunakan lxml. Setelah itu dilakukan perulangan yang akan mencari elemen 'div' yang mengandung kata *"ads"* di kelasnya, dan menghapus elemen tersebut. Setelah itu content yang sudah diolah tadi, dikembalikan menjadi bentuk teks, dan diteruskan ke perangkat client.

```
1  from mitmproxy import ctx
2  from mitmproxy import http
3  import json
4
5  class ReplaceJSON
6    def response(flow: http.HTTPFlow) -> None:
7        if flow.request.url.startswith("http://example.com/"):
8            data = json.loads(flow.response.get_text())
9            data["ads"][0]["advertisement"] = False
10           flow.response.text = json.dumps(data)
11
12 addons = [ReplaceJSON()]
```

Kode di atas merupakan contoh penggunaan *library* JSON, sama seperti kode sebelumnya, kode ini menggunakan *response*. Dalam kode ini jika url yang diakses merupakan "example.com" maka data akan di-*parsing* dengan menggunakan library json. Kemudian data JSON yang dari *advertisement* akan diubah menjadi *False*.

Daftar Pustaka

Cortesi, Aldo et al. (2010–). *mitmproxy: A free and open source interactive HTTPS proxy*. [Version 8.1]. URL: https://mitmproxy.org/.

Dewi, Sari dan Adam Iqbal Islami (2021). 'Implementasi Web Filtering Menggunakan Router Fortigate FG300D'. *INSANtek* 2.1, pp. 22–27. ISSN: 2722-547X.

Google Developer (2022). 'Mengamankan situs Anda dengan HTTPS'. *Google Developer*. URL: https://developers.google.com/search/docs/advanced/security/https?hl=id (diakses pada 17/05/2022).

Habibi, Roni (2022). 'OPTIMALISASI INTERNET WARGA MENGGUNAKAN KOMBINASI TYPE ANTRIAN DAN SISTEM PIHOLE'. *Jurnal Teknik Informatika* 14.1, pp. 1–6.

He, Boyuan et al. (2018). 'An investigation into android in-app ad practice: Implications for app developers'. *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*. Honolulu, HI: IEEE, pp. 2465–2473. DOI: 10.1109/INFOCOM.2018.8486010.

Jin, Ling et al. (2021). 'MAdLens: Investigating into android in-app ad practice at API granularity'. *IEEE Trans. Mob. Comput.* 20.3, pp. 1138–1155. DOI: 10.1109/TMC.2019.2953609.

Lakshmanan, Ravie (2020). 'Hackers Using Google Analytics to Bypass Web Security and Steal Credit Cards'. *The Hacker News*. URL: https://thehackernews.com/2020/06/google-analytics-hacking.html (diakses pada 11/05/2022).

Osborne, Charlie (2019). 'This is how Google Analytics is abused by phishing scammers'. *ZDNet*. URL: https://www.zdnet.com/article/this-is-

how-google-analytics-is-abused-by-phishing-scammers/ (diakses pada 11/05/2022).

S, SARATH, ASIF A dan ARAVIND P (2020). 'Low-cost Security Solution for Micro, Small and Medium Enterprises'. *2020 IEEE International Conference for Innovation in Technology (INOCON)*, pp. 1–9. DOI: 10.1109/INOCON50539.2020.9298273.

Tahaei, Mohammad, Alisa Frik dan Kami Vaniea (2021). 'Deciding on Personalized Ads: Nudging Developers About User Privacy'. *Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021)*. USENIX Association, pp. 573–596. ISBN: 978-1-939133-25-0. URL: https://www.usenix.org/conference/soups2021/presentation/tahaei.

Towidjojo, Rendra (2013). *Konsep dan implementasi routing dengan router mikrotik 200% connected*. Jakarta: Jasakom. ISBN: 978-979-1090-80-3.

Uni Dewi Samima, Helen (2021). 'Perancangan Sistem Blocking Situs Berbahaya Menggunakan Pi-Hole Berbasis Docker'. PhD thesis. Politeknik Negeri Jember.

Wahyudi, Eko, TM Diansyah dan Divi Handoko (2020). 'PEMANFAATAN PI-HOLE DALAM MELAKUKAN BLOK IKLAN PADA WEBSITE DI SMK TIK DARUSSALAM'. *SEMINAR NASIONAL TEKNOLOGI INFORMASI & KOMUNIKASI*. Vol. 1. 1, pp. 263–270.

Yusoff, Siti Intan Nasuha Md dan Shahidatul Arfah Baharudin (2020). 'Virtual Private Network Server and Adblock Server using Raspberry Pi with Parental Control'. *Journal of Computing Technologies and Creative Content (JTec)* 5.2, pp. 88–92.