Analisis Penggunaan Enkripsi Pada Layanan *Proxy Authentication* Squid sebagai *Web Cache* dan *Proxy Server*

Charles Hariyadi (13505105)

Program Studi Teknik Informatika Institut Teknologi Bandung
Jl. Ganesha No.10, Bandung
charles@charles.web.id

ABSTRAK

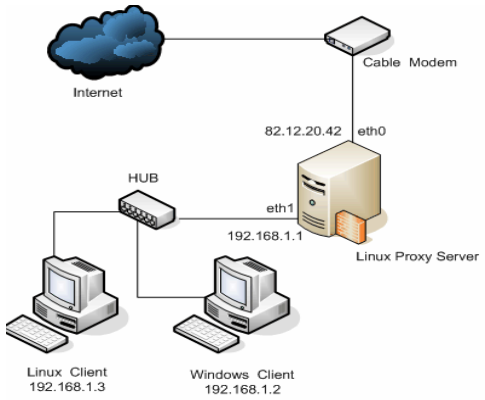
Squid server merupakan suatu *daemon* yang digunakan sebagai *web cache* ataupun *proxy server.* Squid *server* berfungsi untuk mempercepat akses internet dengan melakukan *caching* permintaan terhadap DNS, situs web dan pencarian-pencarian pada web. *Squid server* juga dapat menjadi suatu alat pembantu keamanan dengan cara menyaring lalu lintas yang keluar dari jaringan. *Squid* digunakan pada lingkungan kerja atau kampus untuk melakukan penyaringan jaringan ataupun untuk mempercepat koneksi, sebagai contoh cache.itb.ac.id pada ITB. Untuk melakukan pengawasan dan penyaringan terhadap kinerja internet di dalam lingkungannya, *squid* menerapkan sistem autentikasi. Dalam penggunaannya, *squid* meminta autentikasi dari semua *client* untuk dapat mengakses jaringan internet. *Client* diharuskan memasukkan *username* dan *password* yang telah terdaftar pada *squid proxy*. Setelah *client* memasukkan informasinya, maka informasi itu akan dikirimkan secara langsung ke *squid server* untuk kemudian diautentikasi. Dalam penerapannya, pengiriman informasi berupa *username* dan *password* tersebut tidak terenkripsi, sehingga hal ini memungkinkan terjadinya pengambilan informasi ini oleh pihak yang tidak berhak
Makalah ini akan melakukan analisis tentang penerapan algoritma enkripsi pada pengiriman informasi *squid,* sehingga informasi *client* dapat terjaga kerahasiaannya.

Kata kunci: *Squid*, *server*, *autentikasi, proxy, caching*.

1. PENDAHULUAN

Internet merupakan sebuah sistem global yang dikoneksikan melalui jaringan komputer untuk menghubungkan milyaran pengguna diseluruh dunia. Internet yang berawal dari suatu sistem yang dibangun untuk kebutuhan pertahanan dan keamanan berubah menjadi sebuah alat multifungsi. Internet saat ini menjadi sarana bisnis dan komunikasi yang mampu menghubungkan setiap orang di belahan dunia. Perkembangan internet yang sangat fenomenal ini menjadikan internet sebagai bagian yang sangat tidak terpisahkan dari kehidupan manusia dewasa ini. Kita ambil contoh jaringan sosial *Facebook,* jaringan sosial ini mampu menjadi gaya hidup modern disebagian besar negara. Tetapi dibalik fungsinya yang menarik dan fenomenal itu, internet kerap kali mendatangkan masalah-masalah negatif.

Penggunaan internet yang berlebihan dapat menyebabkan hilangnya fokus kerja dari karyawan perusahaan. Internet juga memungkinkan penyebaran informasi secara bebas, sehingga dapat dipastikan informasi-informasi yang bersifat tidak benar dan berbahaya dapat dengan mudah diakses oleh setiap orang bahkan anak-ank. Karena itulah saat ini dikembangkan berbagai jenis cara untuk melakukan pembatasan terhadap pengaksesan internet. Salah satu cara pembatasan akses dan pemantauan terhadap internet ini adalah *Squid server.*

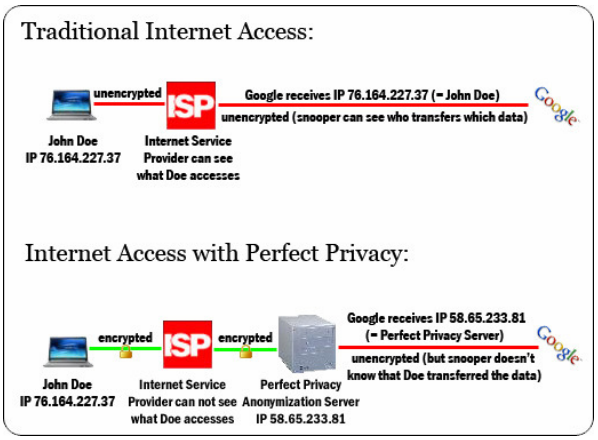


Gambar 1 Arsitektur *client-server* pada penerapan *proxy server*

Squid server merupakan sebuah aplikasi *web cache* dan *proxy server* yang berfungsi mempercepat akses internet dan menyaring serta memantau lalu lintas yang melalui jaringan. Dalam implementasinya *squid server* dibangun dengan arsitektur berbasis *client*-server (gambar 1).

Dimana terdapat sebuah sistem yang berfungsi sebagai *squid server* dan sejumlah *client* yang terhubung dengan *server* tersebut. Dalam penggunaannya, biasanya seorang *client* yang akan mengakses internet diharuskan untuk memasukkan autentikasi berupa *username* dan *password* yang sebelumnya telah terdaftar pada *squid proxy*. Selanjutnya informasi ini akan dikirimkan ke *squid server* untuk diautentikasi, jika informasi yang dimasukkan benar, maka sang *client* berhak melakukan koneksi ke internet. Koneksi yang didapat oleh *client* akan selalu diawasi oleh *rule-rule* yang telah dibuat sebelumnya oleh penanggung jawab *server*.

Autentikasi yang dilakukan oleh *proxy server* biasanya tidak terenkripsi,sehingga memungkinkan pihak-pihak yang tidak berwenang mendapatkan informasi mengenai *username* dan *password* milik *client*. Dengan mendapatkan *username* dan *password* milik *client*, pelanggan yang berada dalam jaringan dapat melakukan akses ke internet tanpa perlu takut identitasnya ketahuan. Sistem autentikasi yang kurang sempurna ini juga mengakibatkan aliran data antar *client-server* yang tidak terenkripsi ini rentan terhadap penyadapan. Untuk itu sistem kriptografi berupa enkripsi terhadap perangkat lunak sangat dibutuhkan oleh *squid server*. Saat ini terdapat beberapa metode autentikasi yang digunakan oleh *squid server,* yaitu metode NCSA *authentication* ,metode PAM, LDAP *authentication, dsb*.



Gambar 2Koneksi yang terenkripsi dan tidak terenkripsi

Tujuan utama dari makalah ini adalah menerapkan autentikasi NCSA *authentication* dan PAM *authentication* pada *squid server* untuk mencegah terjadinya penyadapan data oleh pihak yang tidak berwenang. Tentu saja selain mengatasi masalah keamanan, enkripsi itu harus dapat menghindari masalah waktu yang biasanya dihadapi oleh sistem enkripsi.

2. NCSA *Authentication*

Saat ini perkembangan internet sudah sangat pesat, sehingga sangat mudah untuk melakukan pencurian terhadap *password* milik seseorang yang berada pada jaringan yang sama dengan menggunakan *sniffer tool* biasa. Karena itulah sangat dibutuhkan sebuah sistem autentikasi untuk menjaga keamanan dan kerahasiaan data yang dikirimkan melalui sebuah proxy server.

Terdapat berbagai jenis autentikasi yang dapat digunakan pada sistem *squid server,* tetapi yang paling sederhana dari kesemua sistem autentikasi tersebut adalah NCSA *authentication.* NCSA *authentication* merupakan autentikasi berbasis httpd (web server) *password* yang memungkinkan seorang *client* melakukan koneksi setelah melakukan autentikasi berupa *username* dan *password. Username* dan *password* ini telah tersimpan di server dengan format yang telah ditentukan sebelumnya.
Cara kerja NCSA *authentication* adalah *:*

1. *Client* mengirimkan *username* dan *password* kepada sistem, yang telah terenrkripsi.
2. Sistem akan melakukan decoding ulang dari *password* dan membandingkan dengan berkas *passwd* yang ada pada server.
3. Jika *password* dan *username* cocok, maka *client* akan diizinkan untuk melakukan koneksi internet melalui proxy.

2.1 Enkripsi *Base64*

Dalam penerapannya, *password* untuk NCSA *authentication* merupakan enkripsi berbasis *Base64. Base64* merupakan suatu algoritma enkripsi yang melakukan *encoding* pada data biner dan mengubahnya menjai representasi 64 bit. *Base64* merupakan sebuah algoritma enkripsi yang sederhana dan mudah untuk dipecahkan, tetapi ia memiliki waktu pemecahan yang singkat. Sebagai contoh, kalimat :

"Man is the source of all hatred and love".

Dienkripsikan menjadi :
"TWFuIGlzIHRoZSBzb3VyY2Ugb2YgYWxsIGhhdHJlZCBhbmQgbG92ZQ=="

Tabel 1 Tabel enkripsi *Base64* untuk kata *man*

Text content	M			a				n			
ASCII	77			97				110			
Bit pattern	0 1 0 0 1 1 0 1			0 1 1 0 0 0 0 1				0 1 1 0 1 1 1 0			
Index	19			22			5				46
Base64-Encoded	T			W			F				u

Dapat kita lihat pada tabel 1 diatas hasil enkripsi dari *man* adalah *TWFu*. Jika disandikan menggunakan ASCII, maka

M, a, n akan disimpan sebagai *byte* 77, 97, 110, dimana nilai 2 bit yang didapat secara berurut 01001101, 01100001, 01101110. Ketiga *byte* ini digabungkan menjadi representasi 24 bit yang menghasilkan 010011010110000101101110. Kemudian dilakukan pemotongan 24 bit tersebut menjadi 4 kumpulan bit dengan masing-masingnya berisi 6 bit. Sehingga dapat kita lihat di bagian Index menghasilkan nilai-nilai yang berkorespondensi dengan tabel 64 bit.

Skema enkripsi *base64* biasanya digunakan ketika diperlukan sandi terhadap data biner yang dedesain untuk menangani data berbentuk teks. Hal ini ditujukan untuk menjaga data selama pengiriman ke suatu *server*.

2.2 Penerapan NCSA *Authentication* Pada Squid Server

Karena NCSA *authentication* merupakan metode yang menggunakan *base64* sebagai dasar enkripsi, maka *server* akan menghasilkan *password* dalam bentuk *base64* untuk setiap *user* yang ada. Hal ini dilakukan dengan perintah :

```
# htpasswd /etc/squid/passwd user1
Output:
New password:
Re-type new password:
Adding password for user user1
```

Untuk kemudian dilakukan perubahan terhadap konfigurasi dari *squid server* agar dapat melakukan autentikasi terenkripsi.

```
#  vim /etc/squid/squid.conf
```

Lakukan penambahan potongan kode berikut pada konfigurasi :

```
auth_param basic program
/usr/lib/squid/ncsa_auth
/etc/squid/passwd
auth_param basic children 5
auth_param basic realm Charles proxy-
caching web server
auth_param basic credentialsttl 2 hours
auth_param basic casesensitive off

acl ncsa_users proxy_auth REQUIRED
http_access allow ncsa_users
```

Sedikit penjelasan dari kode-kode diatas :

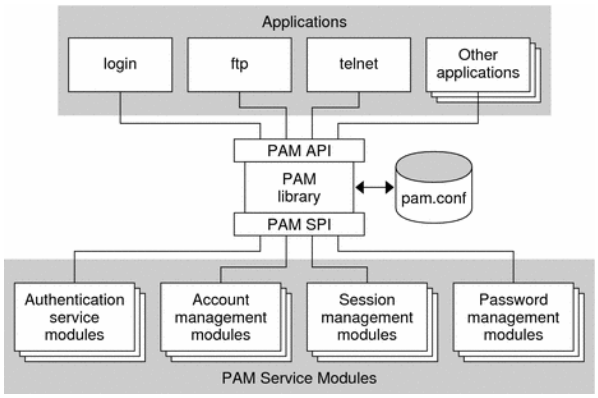
- *Auth_param basic program /usr/lib/squid/ ncsa_auth /etc/squid/passwd* : merupakan spesifikasi file *password base64* dan juga lokasi program NCSA authentication.
- *Auth_param basic children 5* : Jumlah dari proses autentikasi yang akan dilakukan.
- *Auth_param basic realm Charles proxy-caching web server* : merupakan bagian teks yang akan ditampilkan bersama "prompt" untuk meminta *username* dan *password.*
- *Auth_param basic credentialsttl 2 hours* : merupakan jumlah jeda waktu validasi yang akan dilakukan untuk meminta *username* dan *password* dari user. Dalam hal ini, *username* diset menjadi 2 jam.
- Auth_param basic casesensitive off : menspesifikasikan apakah *username* yang dimasukkan bersifat *case sensitive* atau tidak.
- Acl ncsa_users proxy_auth REQUIRED : semua pengguna yang terautentikasi harus memnuhi ACL yang bernama ncsa_users.
- http_access allow ncsa_users : mengizinkan akses terhadap layanan internet jika pengguna berhasil diautentikasi.

Setelah itu dapat dilakukan perintah berikut ini :

```
# /etc/init.d/squid restart
```

3. PAM Authentication

Autentikasi PAM merupakan autentikasi dasar yang tidak berbeda jauh dari penerapan NCSA. Perbedaan utama dari autentikasi ini adalah mengadakan pengecekan komunikasi lebih dari sekali untuk setiap autentikasi yang akan dilakukan.



Gambar 3 *Framework* PAM

3.1 Penerapan PAM Authentication

3.1.1 Server Setup

Konfigurasi squid untuk menggunakan PAM

 # /etc/squid/squid.conf:

```
auth_param basic program
/usr/lib/squid/pam_auth
auth_param basic children 5
auth_param basic realm Squid
proxy-caching web server
auth_param basic credentialsttl 4
hours
acl password proxy_auth REQUIRED
http_access allow password
```

Konfigurasi PAM untuk menggunakan Squid

 # /etc/pam.d/squid:

```
auth required
/lib/security/pam_unix.so
account required
/lib/security/pam_unix.so
This configuration will allow you
to authenticate to the proxy with
a local account.
```

Konfigurasi stunnel for untuk digunakan dengan squid dengan melakukan hal dibawah ini:

```
# openssl genrsa -out privkey.pem
2048
# openssl req -new -x509 -key
privkey.pem -out cacert.pem -days
1095
# cat privkey.pem cacert.pem >>
/etc/stunnel/stunnel.pem
```

Lakukan pengesetan private key:

```
# chmod 0400
/etc/stunnel/stunnel.pem
```

Lakukan pengeditan pada konfigurasi stunnel.

 # /etc/stunnel/stunnel.conf:

```
cert = /etc/stunnel/stunnel.pem
chroot = /var/run/stunnel/
pid = /stunnel.pid
setuid = nobody
setgid = nobody
[squid]
# Ensure the 'connect' line
matches your squid port. Default
is 3128
accept = 8080
connect = 127.0.0.1:3128
```

3.1.2 Client Setup

Konfigurasi Stunnel milik Client

Pastikan stunnel.conf seperti berikut:

```
client = yes

[proxy]
accept = 127.0.0.1:8080
# Replace SERVER with the address
of the server setup previously
connect = SERVER:8080
```

Setelah itu dilakukan pengujian.

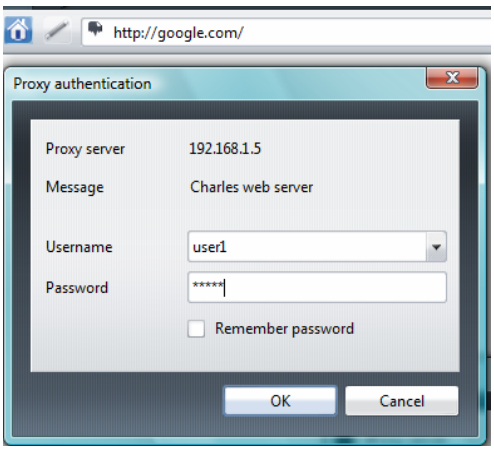
4. Pengujian

Pengujian dilakukan dengan menggunakan :

- *Squid server :* bersistem operasi Ubuntu 8.0, dengan *Squid* v.2.6 (*stable version*)
- *Client :* bersistem operasi Windows Vista Ultimate, menggunakan Opera 10.6

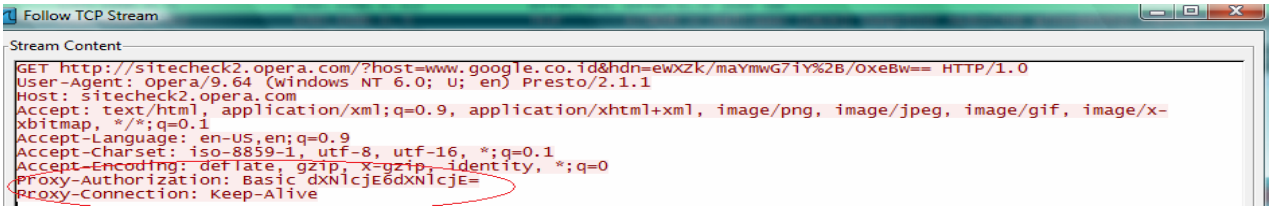
4.1 Pengujian NCSA_Authentication

Pertama akan dilakukan percobaan request terhadap situs *www.google.com* dapat kita lihat hasil yang didapat :



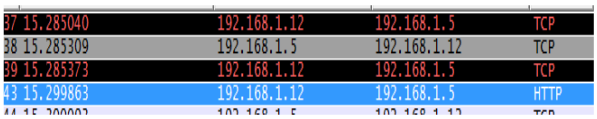
Gambar 4 Autentikasi *username* dan *password*

Setelah *username* dan *password* dimasukkan, maka pada paket *sniffer* akan kita lihat data sebagai berikut.



Gambar 5 Enkripsi terhadap *password*menggunakan NCSA *authentication*

Dapat kita lihat pada gambar 3 bahwa terjadi enkripsi pada *password* yang dimasukkan oleh *user*. NCSA dalam hal ini hanya memberikan satu kali respon terhadap *client* untuk melakukan autentikasi tersebut. Dapat kita lihat pada hal gambar dibawah :

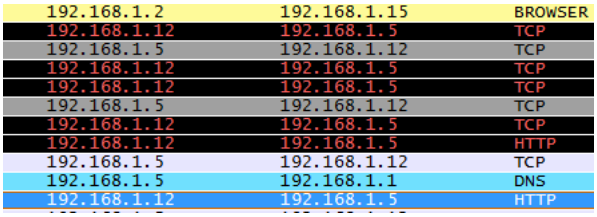


Gambar 6 Hanya terdapat 1 http request

Dalam hal ini hanya dilakukan sekali komunikasi antar *client –server* untuk melakukan autentikasi dari pengguna. Waktu yang dibutuhkan untuk melakukan autentikasi juga tidaklah lama. Dapat dilihat pada gambar 6, waktu yang dibutuhkan oleh *client* untuk menyelesaikan autentikasi adalah 0.01 detik.

4.2 Pengujian PAM *Authentication*

Pengujian PAM tidak berbeda jauh dengan pengujian menggunakan NCSA *authentication.* Perbedaan utama dari pengujian ini adalah *request* yang berulang-ulang terhadap server untuk melakukan konfirmasi dan *tunnelling.*



Gambar 7 Request yang berulang-ulang terhadap server

Dari gambar 7 diatas dapat kita lihat bahwa client (192.168.1.12) melakukan request yang berulang-ulang terhadap *server* (192.168.1.5). Hal ini menyebabkan terjadinya tenggang waktu sedikit lebih lama dari saat menggunakan NCSA *authentication. Request* yang dilakukan oleh *client* adalah autentikasi berulang yang merupakan ciri khas dari PAM.

5. Kesimpulan

NCSA *authentication* merupakan sebuah sistem enkripsi yang sangat mudah diterapkan untuk *squid server.* Tetapi masih terdapat beberapa kekurangan yang harus diperhatikan dalam penggunaan *autentikasi* jenis ini untuk mengamankan pengiriman *password*

Tabel 2 Tabel perbandingan PAM dan NCSA

Perbedaan	PAM	NCSA
Jumlah Proses	Selama Aplikasi berjalan	Sekali autentikasi
Waktu autentikasi	0.05	0.01 detik

NCSA merupakan autentikasi yang mengutamakan kesederhanaan. Sehingga autentikasi hanya dilakukan diawal dan dapat dilakukan berkali-kali oleh pengguna ditempat dan waktu yang berbeda. Untuk selanjutnya diserahkan pada *credentialttl* untuk menentukan waktu hidup dari autentikasi tersebut. Karena kedua metode ini masih menggunakan metode *password* httpd *basic.* Jadi pengiriman *password* masih berupa teks yang berisi hasil enkripsi, sehingga masih terdapat kemungkinan pengambilan *password* oleh pihak yang tidak berwenang, walaupun sedikit. Kelemahan terakhir NCSA adalah penggunakan *base64* yang merupakan algoritma enkripsi standar. Algoritma ini merupakan algoritma yang sangat mudah dipecahkan untuk saat ini, misalnya :
Terdapat *username* dan *password :*

```
Fannie:FuRpAnTsClUb
```

Dengan hasil enkripsi :

```
Authorization:Basic
mFubmllOkZ1UnBBblRzQ2xVYgo=
```

Maka untuk melakukan dekripsi di linux cukup menggunakan perintah

```
% echo RmFubmllOkZ1UnBBblRzQ2xVYgo= |
/usr/local/lib/python1.5/base64.py -d
```

Didapat
Fannie:FuRpAnTsClUb

Karena itu metode ini masih rentan terhadap penyerangan. PAM walaupun merupakan autentikasi basic seperti NCSA, tapi karena menerapkan banyak *layer* autentikasi, PAM tergolong aman. PAM juga jarang menggunakan algoritma *base64* untuk melakukan enkripsi *password.* Berikut hal-hal penting mengenai PAM :

- Menggunakan enkripsi selain DES, sehingga menyusahkan penyerangan.
- Terdapat *password* bayangan.
- Mengizinkan seorang user melakukan autentikasi pada satu saat dan satu tempat.

Dalam dunia nyata menggunakan enkripsi untuk sistem *Squid server* tidak menjadi solusi yang menyelesaikan masalah keamaan data pelanggan. Penyadapan dapat terjadi di manapun. Kakas yang digunakan untuk serangan juga semakin mudah dicari. Sehingga perlu digunakan keamanan berlapis yang melibatkan seluruh komponen jaringan. Untuk lebih meningkatkan keamanan, teknik enkripsi ini sebaiknya dipadukan dengan firewall yang akan menyaring aliran data sehingga penyadapan data dapat dihindari. Penerapan enkripsi untuk *squid server* masih jauh dari sempurna, karena itu diperlukan lagi pengembangan lebih lanjut mengenai sebuah system enkripsi yang mampu meberikan enkripsi yang kuat tanpa menghabiskan banyak waktu.

REFERENSI

[1] Wessels Duanne , "Squid The Definitive Guide", O'Reilly, 2004.

[2] "*Password Security and Encryption*", http://www.nic.com/~dave/SecurityAdminGuide/SecurityAdminGuide-9.html, 2000.

[3] Stallings, William *Cryptography and Network Security Principles and Practices, Fourth Edition.* Prentice Hall 2005

[4] Khurana, Himanshu, Koleva, Radostina and Basney, Jim "Performance of Cryptographic Protocols for High-Performance,High-Bandwidth and High-Latency Grid Systems, National Center for Supercomputing Applications (NCSA),2008.

[5] *Squid faq,* http://www.comfsm.fm/computing/squid/FAQ-23.html