Global Information Assurance Certification Paper

Practical for GIAC Certification

Understanding and implementing socks server – A guide to set up a socks environment.

Name	Carsten Thieswald
Assignment versionnumber	1.4b
Certification grade Option	GSEC 1

Table of contents

1 Introduction

This paper covers a very interesting tool regarding network security: the socks server. It is intended to provide an understanding of the topic and then use these acquired skills in a sample case.

It starts with an explanation of the basic functions of proxy servers, from which it derives the functionality deployed at the socks server. Then the two protocol implementations of version 4 and version 5 are compared. After providing a basic knowledge it goes more into detail regarding advantages of socks server environments. A socks server requires a counterpart at the client side. Some different client approaches will be shown in this paper. The second part of this paper is dedicated to the practical use of socks server. First of all two socks server implementations are presented. This is followed by a description of 5 socks client implementations. Finally everything is deployed to a realistic scenario. To this specific requirement socks server and client configuration are developed.

2 Proxy Server

Peter Grennan summarizes the function of a proxy server in his firewall introduction in the passage Proxy Servers: “Proxies are mostly used to control, or monitor outbound traffic. Some application proxies cache the requested data. This lowers bandwidth require ments and decreases the access time to the same data for the next user.”[16] The following drawing describes the function more in detail.

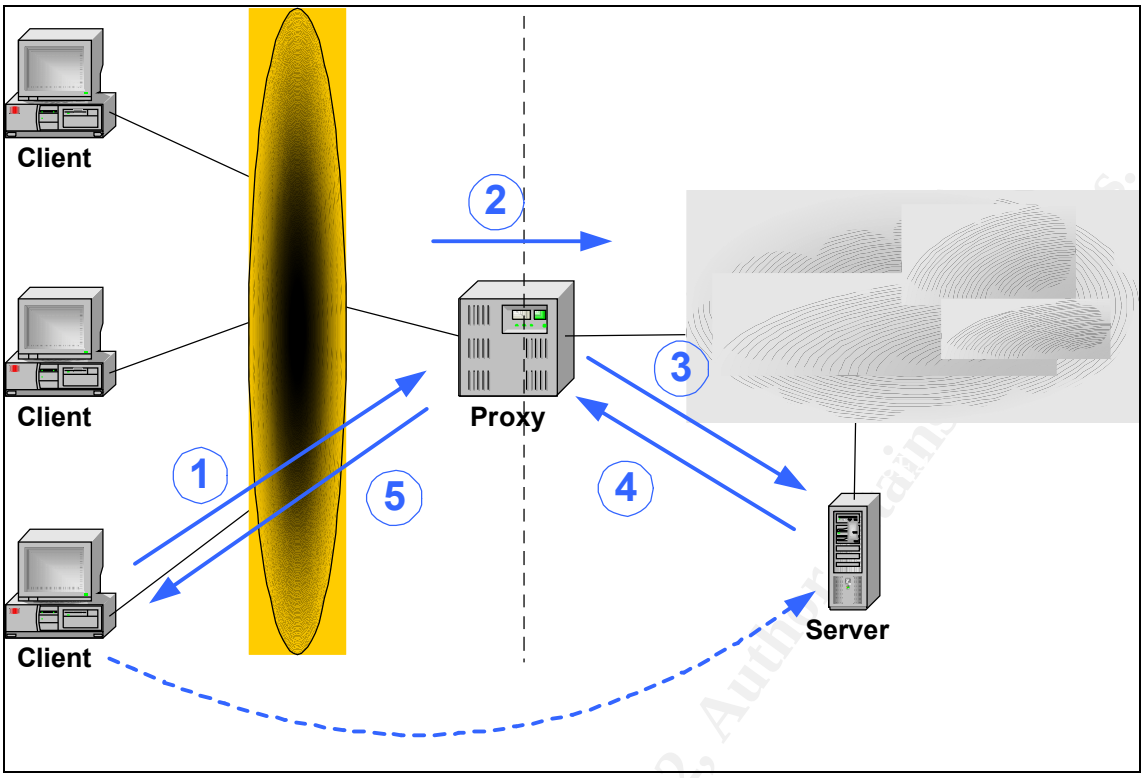


Figure 1: How a proxy server works

1) Client contacts proxy in stead of the originally addressed server
2) Proxy evaluates client request and decides which to pass or which to reject
3) Proxy transfers request to originally addressed server on behalf of client
4) Server answers to proxy without knowing about client behind
5) Proxy passes answer to requesting client

Clients cannot decide whether to communicate to a server directly or via a proxy. They are forced to use the proxy by means of configuration in the corresponding application. On the other hand the application server only knows about the proxy and is supposed to only communicate with it. Proxies are also used as a kind of firewall because they protect the clients against contacts from outside. Clients to the left of the proxy are unknown to hosts on the right side (se e drawing).
This fact can be used to reduce the requirement of official ip addresses. When using a proxy server one single official ip address is theoretically sufficient to access the Internet for a whole company. The proxy keeps a table of all sessions and connections. It maps the ip addresses and port numbers from inside to a single ip address and the corresponding port number. This function is called NAT (Network Address Translation).

“An example for a proxy is a person telnetting to another computer and then telnetting from there to the outside world” as Mark Grennan describes it in his “Firewall and Proxy Server HOWTO chapter 11 URL:” http://www.tldp.org/HOWTO/Firewall -HOWTO-2.html [17]. A proxy server that automates this process is called transparent proxy. Because proxy servers handle all network communication, they can log a lot. For HTTP proxies this includes requested URL’s. For FTP proxies this includes every file that is transfe rred. They can even filter out "inappropriate" words and block sites from which the URL was retrieved or scan for viruses. [17]

There are two major types of proxy servers. The first one is called dedicated or application level proxy and the other one gene ric or circuit level proxy. Dedicated proxies are specialized on one or a small amount of protocols. They know in detail about the mechanism of the proxied service and can check the contents of the packets. The name application level indicates that the pro xy can work on the application layer of the TCP/IP stack. [1],[3]

Generic proxies are more general proxies. In contrary to their specialized colleagues they can be called all-rounder. They cannot look beyond the port number. This means they blindly trus t the destination port number of a packet. When there is something addressed to the HTTP port 80, the generic proxies treats it like a HTTP request without further checking. This may also be a disadvantage. Generic proxies can easily be faked. You can sen d SMTP traffic via port 80 and the proxy will pass it although smtp is not allowed. There are some more complex protocols like ftp that don't function in combination with generic proxies because they need special treatment.[3]

3 Socks Proxy Server

The socks proxy server is a generic proxy and supports nearly every application. It only requires the application to run socksified. There are currently two socks standards implemented: Socks Version 4 and the more sophisticated Version 5. Both protocols are n ot compatible but socks5 understands socks4.[3]

Socks Version 4 provides a mechanism for TCP based traffic to transparently connect to an application server via a socks gateway. Based on destination and source address respectively port number the access to the application server is granted. The only authentication applied is identd. There are two disadvantages in the socks4 implementation:[3]

- No or weak authentication
- To make a client applicaton socksified it has to be compiled against the socksV4 library

Socks Version 5 has the following major improvements:

- User authentication
- Name resolution
- Support of UDP packets

Socks4 acts like a packet filter that grants access by means of access control lists (ACL's) based on ip addresses and port numbers. Addition ally socks5 provides different types of user authentication methods. The client propagates the methods it supports and the server decides based on its security policy which one to choose.
In early socks4 the name resolution was done at the client's side. S ervers which names could not be resolved locally were unreachable when addressed by name. This changed in Version 5. Now it is possible to pass a name to the socks server that resolves it on behalf of the client. The socks server then asks his name server.
Prior to Version 5 UDP packets were not supported, this means they were dropped. Now applications based on UDP like i.e. DNS and NFS can communicate through socks proxy server. [3]

As the following drawing illustrates a socks server listens by default on port 1080. Applications that want to use the socks server have to be socksified. The socks client wraps the original request into a socks request. This means all outbound traffic is redirected to the socks server at port 1080. The socks server evaluates t he original request and passes it to the destined server if allowed. The source address will be replaced by the socks servers own and the source port

will be randomly chosen by an available high port of the socks server. Now let's have a look at the advantages and disadvantages the use of socks servers can have.

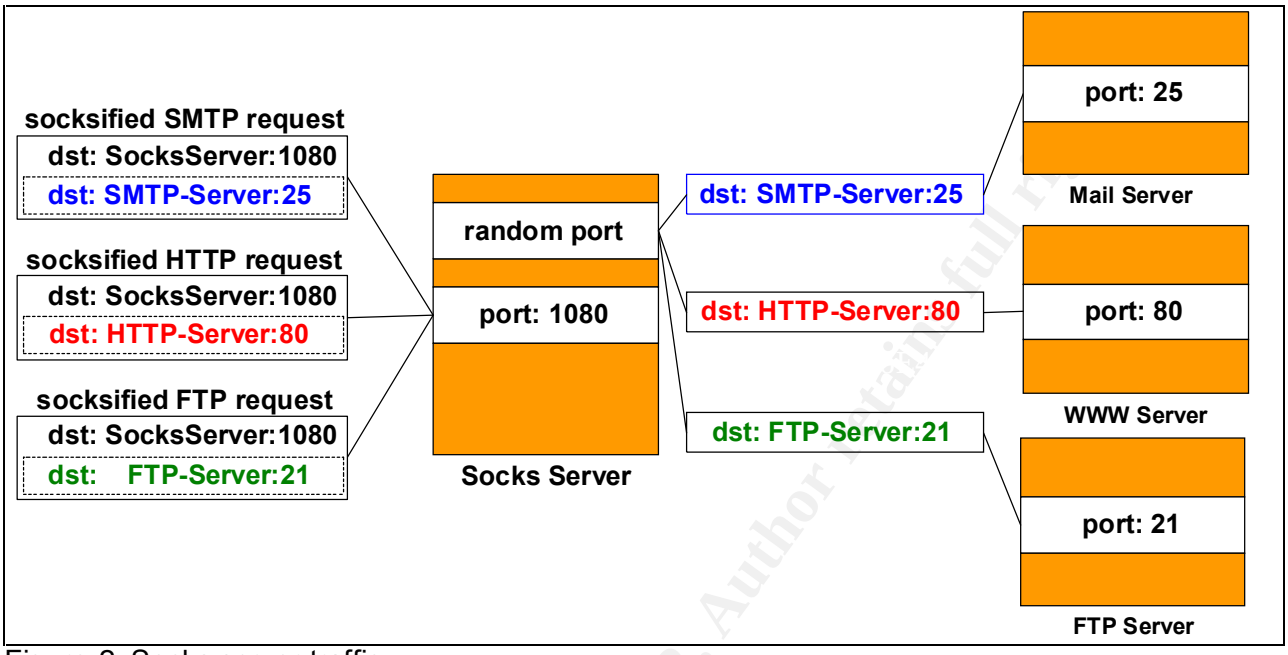


Figure 2: Socks server traffic

Advantages of socks server:

- Usually a socks server listens at one port which is by default port 1080. This is used by all socksified applications. To look only for one port facilitates management and monitoring of network traffic.
- Establishment of new socks sessions follows the TCP respectively UDP protocol conventions. In combination with user authentication this provides a rather strong level of integrity. There are other solutions where user authentication is done separately from session establishment like client authentication at a firewall. A client connects i.e. to a firewall for authentication. After a successful username and password exchange the user is granted access depending on his privileges. The firewall often opens a tunnel to the user's ip address to provide the granted access. It's easier to take over this tunnel than a socks session. Independe nt of the described solution it is better to do user authentication in relation with session establishment in a single process.[13]
- The use of socks server is independent of the application because of the generic approach of socks. New applications can be run immediately without any need of customization. There are applications that may have problems when running socksified. But this is often caused by not using the given standards i.e. passing by the operating systems tcp stack.[13]
- A very granular access control mechanism is applied by means of packet filters. For inbound connections the ip range of clients that are allowed to connect to the socks server can be limited. In conjunction with user authentication this can be restricted to a single user. The ou tbound ACL specifies to which server and services dedicated clients are allowed to connect. All these activities can be logged. [13]
- A socks server provides a possibility for inbound connections something that general internet proxies don't offer. The oper ation to establish outbound connections is called CONNECT. A client connects to the socks server and sends a CONNECT request with target address and port number. The socks server evaluates the request and makes a connection if allowed. After a successful C ONNECT it can be nesessary that the internet

server needs to be connected directly to the client. This will be handled on the socks server by an operation called BIND. The client connects again to the socks server and sends a BIND request and includes the internet server's ip addressed and port number. If allowed the socks server reserves a socket and waits for an incoming connection. All traffic will be transparently passed between the internet server and the client.[13]

- Several socks server can be used i n a chain or in the mode socks chaining. A socksified packet that arrives at a socks server is usually unwrapped and sent directly to the application server. When a rule on a socks server says that the target destination can only be reached via an addition al socks server the packet will be redirected to the second socks server. The packet stays wrapped and the protocol used is socksv5. By applying encryption socks chaining can be used as a VPN. [19]

Disadvantages of socks server:

- When deciding to use socks server a socks environment has to be established. All clients have to be customized. The configuration files have to be maintained. An introduction has to be planned carefully.
- Like all circuit level proxies the socks server proxy faces the same we akness. Access control is limited to ip address, port number and user authentication. The contents of a packet can't be evaluated something an application level proxy could do. This can be used to abuse a socks server by using i.e. http as destination port and sending different traffic that at another port would have been blocked.

4 Socks Clients

An application doesn't connect automatically to a socks server. The operating system or the application itself has to be modified to use socks. This can be achi eved in two general ways. The application has to be compiled against the socks libraries or a socks library has to be made available at runtime. To run an application socksified means to redirect standard network calls to the corresponding programs provide d by the socks library. There are a lot of possibilities in doing so. [3]

- Socks-capable applications such as web browsers, ftp, Citrix or irc clients have an optional socks functionality. When selecting this option the socks server address and the port number socks is listening to, has to be provided. The socks libraries are included in these applications.[19]
- Single windows applications can be socksified by means of programs like SocksCap from NEC [23]. It intercepts the networking calls from winsock appl ications and redirects them through the socks server [15]. In the SocksCap setup the socks server has to be defined. To run a single program socksified it has to be inserted in the SocksCap program list and started from there.
- A whole windows client can b e socksified by manipulating the TCP stack. This is usually done by a socksify program or more common called a socks client. It replaces parts of the TCP stack or introduces an additional layer to the stack. When an application requests a network connectio n the destination ip address is checked against the socks client's configuration. If an entry is found the request will be redirected to the corresponding socks gateway.[19] [3]
- In the Linux/Unix environment socks libraries are available for free. To make an application socks capable it has to be run with the according socks libs. You can either recompile the application against the shared libraries or load these libraries at runtime by starting the application with programs like socksify or runsocks. [3]

5 Dante Socks Server Implementation

After summarizing general features of socks servers and clients we will look at some implementations. The first one is a server called Dante from Inferno Nettverk A/S, a norvegian company. Dante is usually part of the Inf erno Nettverk firewall system but also made separately available via ftp.[5]

There is a detailed instruction about the installation of the software included. For most purposes it is sufficient to deploy the standard commands *configure*, *make* and *make install*. The Dante source packet provides the possibility to generate RPM's. A prerequisite is that the rpm -build packet is installed. For automatically compiling ldap support the ldap -devel packets have to be installed as well. Then the RPM's can easily be cr eated :

```
rpm –ta dante.xxx.tar.gz
```

Server configuration is stored by default in /etc/sockd.conf. The dante man pages offer a detailed description of all features. I would like to mention some of general interest.

First we start with the global server settings before we take a closer look at the access rules. There are two types of rules that have to be defined in sockd.conf. Client rules are mainly designed to specify which client is granted access to which socks server. In the socks rule section the actual application request is evaluated. By adding port numbers or ranges access to specific hosts or networks can be limited.[4]

Keyword	Description
Global Serversettings	
Logout	Where to send the logoutput. Following options are available: syslog, stdo ut, stderr or filename. i.e.: *logoutput: syslog*
Internal	Ip address or interface the socks server is listening and accepting connections i.e.: *internal : 192.168.206.128 port = 1080*
External	Ip address or interface used for outgoing connections i.e.: *external : 192.168.207.128*
Clientmethod	List of acceptable authentication methods for socks -rules (none, rfc931 and pam) in order of preference. These methods are based on clients TCP connection. Methods not listed here can't be used in the socks -rules section
Method	List of acceptable authentication methods for socks -rules (username, none, rfc931 and pam) in order of preference. Methods not listed here can't be used in the socks -rules section. i.e. *method : username none*
Client rules: client pass/blo ck	
From to	Specifies range of clients that are allowed to connect to socks server i.e. *from: 10.0.0.0/8 port 1 -65565 to: 192.168.206.128*
Log	Specifies what to write into logfile. i.e. log: connect error
Socks rules: pass/block	
From to	See above
Method none	No authentication required
Method username	Username/password required. Checked against local passwd file.
Method rfc931	Clients identd provides username which must match local passwd file
Method pam	Pam module selected for user authentication

When building RPM's a start script is provided that is stored under "/etc/init.d/sockd". A start command should start similair to

/usr/sbin/sockd –D –f /etc/sockd.conf.

The configuration can be checked before starting the daemon by submitting the command :

sockd –V

with the corresponding config file.

6 NEC's Socks Server implementation

NEC provides a commercial socks server. It is part of their product called e -Border. But they also publish a socksv5 reference implementation to validate the socks ver sion 5 protocol, and to serve as a framework for inter -operability testing for products based on the socksv5 protocol.

Socks reference software is available free of charge for non commercial use only (academic, research, and personal use). All other usage requires licensing.[24]
It's rather easy to install on a Linux system: configure && make && make install.

The configuration of the socks server is by default stored in /etc/socks5.conf. It mainly consists of 4 sections which will be explained in the follo wing table. The man pages offer a more detailed explanation.

Section	Description
Auth	Type of authentication that can be used Syntax: Auth src-host src-port auth-method
Interface	It is used on multi-homed systems. Networks or services are forced to dest ined interfaces, i.e. inbound traffic is allowed on eth0 and outbound on eth1. Syntax: Interface hostpattern portpattern int -addr.
proxy-type [socks5,socks4,noproxy]	Used for proxy chaining, when the destination network can only be reached via another p roxy. Syntax: Proxy-type dest-host dest-port proxy-list
Access control [permit,deny]	The access control list determines under which conditions an establishment of a requested connection will be permitted or denied. Syntax: Permit auth cmd src-host dest-host src-port dest-port [u-list]

7 Hummingbird Socks Client

The Hummingbird Socks Client is running on windows platforms. It comes with a setup program. After installation the socks client has to be configured by modifying the file socks.cnf. I'd like to focus on some important entries of this config.[7]

Keyword	Description
BIND-MODULE	Specifies which applications expects inbound connections. On windows computers it's usually only FTP
EXCLUDE-MODULE	Specifies which applications will not use socks rules. They will connect directly to the destination.
PROXY_NAME	Specifies servers to use for name resolution if local lookup fails.
DIRECT dst_addr dst_mask	Specifies the net to which you will be connected directly.
SOCKD4 @=serverlist dst_addr dst_mak	To reach the net dst_addr you have to use one of the socks servers specified in serverlist. The protocol is socks V4.
SOCKD5 @=serverlist dst_addr dst_mak	According to line above.
BALANCE SOCKD5 @serverlist … …	This keyword is used for loadbalancing. The soc ks server in serverlist will be used randomly.

8 Aventails Socks Client

There are several versions of Aventail Socks clients available that run on windows. Recently they came out with Version 5. I would like to go more into detail with version 3 because I 'm still running this and all important features are included. Aventail is a windows socks client and comes with a setup program. The configuration differs from other clients. There is no config file in ascii text. A special configuration program is provid ed to create or modify a configuration. It consits of maps that contain the necessary information which will result in socks rules in the map redirecton rules.

Map	Description
Servers	Specifies a socks server: - Hostname or IP-Address - Portnumber - Protocolversion - Fallbackserver
Destinations	Specifies destination to reach: - Hostname or IP-Address - or network - or DNS domain
Advanced	- enable redirection through successive socks servers - List of application to use socks rules
Name Resolution	- Contains list of known domains - Specifies which socks server to use for remote name resolution
Redirection Rules	Contains a list of socks rules that consist of the following three entries: a) One Destination from the destinations map b) In services are listed used ports c) The proxy redirection offer the three possibilities to redirect via a socks server, to not redirect or to deny

9 Socksify using Sockscap

Sockscap is a windows program that enables you to run single windows programs in a socksified environment. It can be compared to running a shell. Every application that should run socksified has to be inserted in the Sockscap applications list with its corresponding command line and working directory. A general setup is to be done to initialize the socks environment. The features are listed in the following table:

Feature	Description
Servers	Specifies a socks server - Hostname or IP-Address - Portnumber - Socks5 Userid
Protocol	- Sock4 or Socks5
Supported authentication	- GSSAPI - Username/password
Name Resolution	- Resolve all names locally - Resolve all names remotely - Attemp local than remote
Direct connections	Specifies destinations to which connect directly

Unfortunately not every application runs socksified with SocksCaps. It's not aimed to be a tool for general use. But it can be useful in environments, where single applicaltions have to use a socks gateway.

10 Socksify using client implementation of Dante

This is the client part of the Dante socks server that runs on Unix or Linux systems. There are two modes of running the socks client. The first one is called socksify on demand. This means the socksify command will be followed by the intended command, i.e.

```
socksify telnet 192.168.100.123
```

Executing commands with "socksify" is like executing them in a different shell. In this case different libraries will be used. The second mode to run socks clients is to create a permanent socksified. This will be done by loading the corresponding libraries. In a bash shell on a RedHat system you have to set a variable as follows:

```
export LD_PRELOAD="/usr/lib/libdl.so /usr/lib/libsocks.so"
```

Now all applications will run socksified according to the configuraion.

The configuration of the socks client is stored by default in /etc/socks.conf. The man pages of socks.conf offer a detailed explanation of all features. There are also some example configurations in the source packet. I'd like to discuss a simple config to get easily started.

```
# 1.
route {
from: 10.0.0.0/8 to: 10.0.0.0/8 via: direct
}
# 2.
route {
from: 10.0.0.0/8 to: 192.168.12.0/24 via: 10.1.100.1 port = 1080
}
# 3.
route {
from: 10.0.0.0/8 to:  0.0.0.0/0  via: 10.1.1.1  port = 1080
}
# 4.
route {
from: 10.0.0.0/8 to:  .  via: 10.1.1.1  port = 1080
}
```

1. Clients on the local network don't use a socks server. Connections are established directly.
2. The class C net 192.168.12.0 will be reached via the socks server 10.1.100.1. A named destination would be possible as well.
3. All other directions will be routed through a different gateway
4. Means the same as 3 and is used when there is no local name resolution

The Dante socks client is a very powerful implementation. It's possible to direct to http - and ms-proxies as well. Both socks protocol versions are supported. But there is no option for fall back or high availability. If there were to be a redundant socks server it couldn't be addressed. The only way would be to use a name for the socks server and store both ip addresses for this name. Then the DNS server can act as round robin and can provide alternating ip addresses. But in this case socks functionality depends on the availability of a local name server.

11 Socksify using client implementation of NEC

The NEC socks client is a UNIX based implementation. It is provided with the server packet and has a similar syntax. To run a command stri ng socksified it's necessary to submit "runsocks" befor it. This will cause the socks libraries to be loaded dynamically and a socksified environment to be created. It's like a socksified shell for this single command.

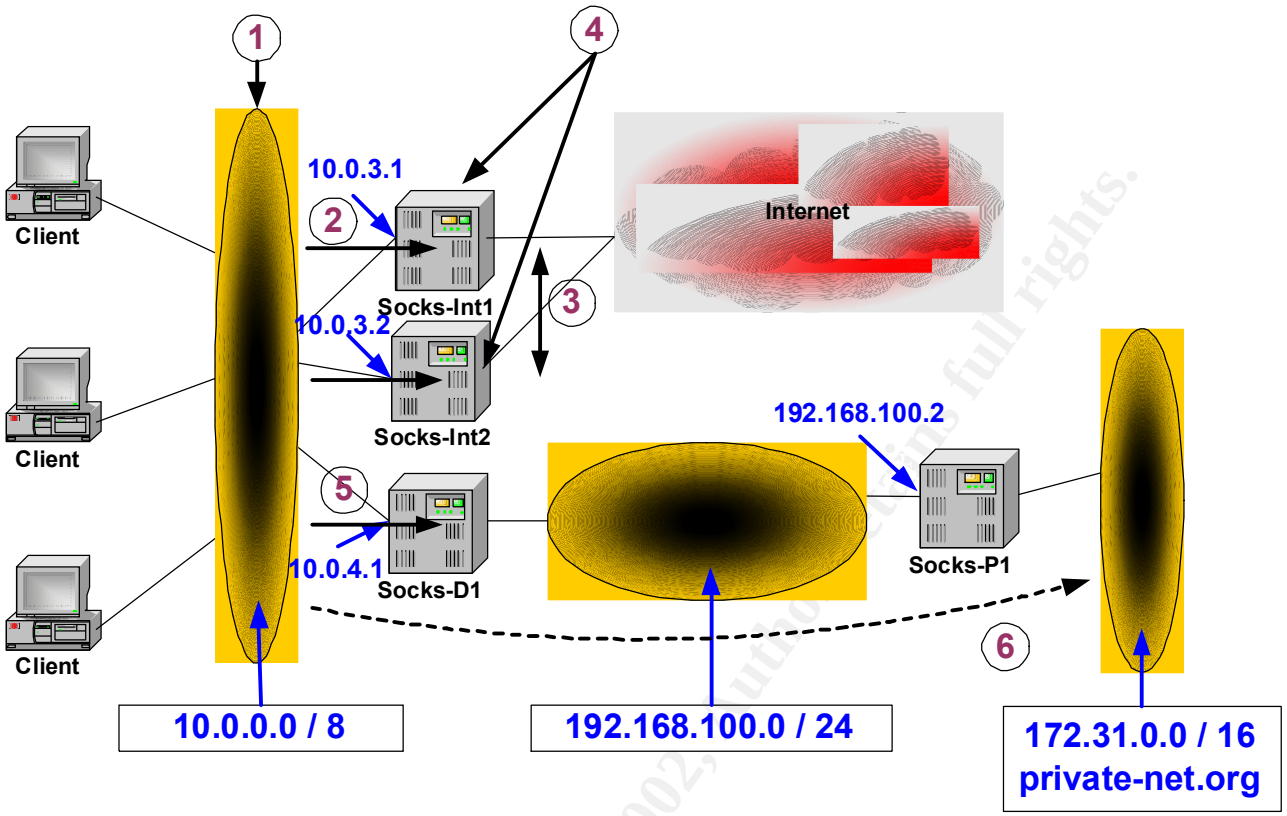
The configuration of the client mai nly resides in /etc/libsocks5.conf. It contains information whether and how the client should connect to the specified socks server. The entries have the following syntax:

```
Proxy cmd dest-host dest-port [user-list [proxy-list]]
```

Proxy is the the communica tion protocol like socks4 or socks5.
Cmd is a command like connect or bind.
Proxy-list is a list of socks server or just a single one via which the destination can be reached..

12 A Scenario

To put all together what was mentioned in this paper we will ap ply it to a realistic scenario like the following drawing demonstrates. A company uses generously the whole private class A net 10.0.0.0/8. It accesses the internet via socks servers version 5. Only outbound http will be allowed and user authentication wil l be required. To increase the availability two socks servers are identically configured. If possible they both should be activly used. A subsidiary of our company is provided with the subnet 192.168.100.0/24. Access to this net is controlled by Socks-D1. There is another net that can be reached via the subsidiary. The domain names of this net can't be resolved in the companies local net. Therefore we use the feature of socks5 to resolve the names on the remote socks server. This means the packet will be s ent directly to the socks server that on his side asks his name server for the corresponding ip address. To pass the subsidiaries net we have to deploy proxy chaining. The client only knows about the first socks server Socks-D1 and addresses this one to re ach private-net.org. Socks-D1 redirects all requests to private-net.org to Socks-P1. In this example we only use socks version 5.



In detail we have the following requirement:

1) All destinations at the private net 10.0.0.0/8 will be reached directly
2) The Internet will be reached via the socks gateways Socks -Int1 and Socks-Int2. Only HTTP will be allowed.
3) The two internet socks gateways are redundant.
4) User authentication for access to the internet is required.
5) Destinations at 192.168.100.0/0 are accessible via Socks -D1.
6) Destinations at .private-net.org can be reached via Socks -P1. This includes socks chaining and remote name resolution.

Sample configuration of Dante's socks server:

The configuration of Socks-Int1 and Socks-Int2 are rather similar. Therefore we just take into consideration the first one. Here are the entries of /etc/sockd.conf.

```
# /etc/sockd.conf of Socks -Int1
# GLOBAL SETTINGS
logoutput:   syslog
internal :     10.0.3.1 port = 1080
external :     x.x.x.x
clientmethod : none
method :      username,none
# CLIENT SOCKS RULES
# 1) allow whole Class A net to connect to the socks server Socks  -Int1
client pass {
        from: 10.0.0.0/8 to: 10.0.3.1
        log: connect error
        }
# block anything else
client block {
        from: 0.0.0.0/0 to: 0.0.0.0/0
        log: connect error
        }
# SOCKS RULES
# 2) pass only http requests
# 4) userauthentication local on socks server, checked against /etc/passwd
pass {
        from: 10.0.0.0/8 to: 0. 0.0.0/0 port = http
        method: username
        log: connect error
        }
# block anything else
block {
        from: 0.0.0.0/0 to: 0.0.0.0/0
        log: connect error
        }
```

In the configuration we start with the global settings . We specify where the logoutput is written to, we limit inbound and outbound traffic of the dual homed server to it's according interfaces and define username as authentication method. Access of the whole class A net to the socks server is defined in the clients socks rule section. HTTP access is defined in the the socks rule section. The used authentication method is username. As long as no users are defined, the authentication is checked against the local mechanism i.e. /etc/passwd /etc/shadow.

```
# /etc/sockd.conf of Socks-D1
# GLOBAL SETTINGS
…..
# CLIENT SOCKS RULES
…...
# SOCKS RULES
# 5) pass any request to 192.168.100.0, no user authentication
pass {
        from: 10.0.0.0/8 to: 192.168.100.0/24
        log: connect error
        }
# 6) Destinations at .private-net.org can be reached via Socks -P1
pass {
        …
        }

# block anything else
block {
        from: 0.0.0.0/0 to: 0.0.0.0/0
        log: connect error
        }
```

In the Socks-D1 configuration we only allow requests to 192.168.100.0/24. But the range of services is not limited. All destination ports are allowed and no authentication is required. Socks chaining is unfortunately not yet implemented in Dante's socks server (version 1.1.13). It's on the developer's todo list but not yet implemented. The net private -net.org can't be reached directly by means of chaining. A workaround would be an additional server in the net 192.168.100.0/24 that serves as a stepping stone. Using Aventail socks clients will also solve this problem. This socks client provides on the advanced tab of the config tool an option called "enable rediection through successive socks server". The client config will contain among others two rules. One says you can reach private -net.org via Socks-P1. The second rule tells the client that it can reach 102.168.100.0/24, the net Socks -P1 belongs to, via Socks-D1. Successivly the client applies the two rules with the same result as socks proxy chaining. Because these solutions don't provide any new information , we pass the configuration of Socks -P1.

Sample configuration of NEC's socks server:

The configuration file looks in relation to Dante's implementation more compact and less readable. On the other hand when you get accustomed to them, it is very smart.

```
# /etc/socks5.conf of Socks -Int1
#
interface 10. – 10.0.3.1
interface  – – x.x.x.x
auth – – u
# 1) allow whole Class A net to connect to the socks server Socks  -Int1
# 2) pass only http requests
# 4) userauthentication local on socks server
permit u – 10. – – 80
```

We bind all requests from 10.0.0.0/8 to 10.0.3.1. This means all outbound traffic is bound to this interface. All other traffic is bound to the external interface with the official ip address x.x.x.x. The offered authentication method is user. How to enable Username/password authentication is described at NEC "http://www.socks.nec.com/userpass.html". The last line permits access from 10.0.0.0/8 to everywhere at port 80. User authentication is required.

```
# /etc/socks5.conf of Socks -D1
#
interface 10. – 10.0.4.1
interface  – – 192.168.100.1
auth – – –
# 6) destination private -net.org can be reached via Socks -P1
socks5 .private-net.org – 192.168.100.2
# 5) allow whole Class A net with dest ination 192.168.100.0 to connect to the socks server Socks-D1
permit – – 10. 192.168.100. – –
```

At this server 10.0.4.1 is the inbound interface and 192.168.100.1 the outbound. No authentication is required. This line could have been omitted. Traffic to pr ivate-net.org is redirected to the next socks server. This is called socks chaining. Name resolution will be done at the destination socks server. All traffic from 10.0.0.0 with the destination 192.168.100.0 is allowed.

```
# /etc/socks5.conf of Socks -P1
#
interface 192.168.100. – 192.168.100.2
interface  – – 172.31.0.1
auth – – –

# 5) allow 192.168.100.0/24 net with destination 172.31.0.0 to connect to the socks server Socks-P1
permit – – 192.168.100. .private -net.org – –
```

Inbound and outbound interfac es are are defined. The only rule is to allow requests from 192.168.100.0/24 to private -net.org. Socks-P1 knows about a name server that can resolve the domain names.

Sample configuration of Hummingbirds socks client:

Now we'll take a look at the client' s side. We start with the windows implementation of Hummingbird. The configuration resides in socks.cfg.

```
# sample configuration of Hummningbirds socks client
# 1) All destinations at the private net 10.0.0.0/8 will be reached directly
DIRECT 10.0.0.0 255 .0.0.0
PROXY_NAME @=10.0.3.1,10.0.3.2
# 5) Destinations at 192.168.100.0/0 are accessible via Socks -D1
SOCKD5 @=10.0.4.1 192.168.100.0 255.255.255.0
# 6) Destinations at .private -net.org can be reached via Socks -P1
SOCKD5 @=10.0.4.1 *.private -net.org
# 2) The Internet will be reached via the socks gateways Socks -Int1 and Socks-Int2 (http).
# 3) The two internet socks gateways are redundant.
BALANCE
SOCKD5 @=10.0.3.1,10.0.3.2 0.0.0.0/0 0.0.0.0 80
```

All servers in 10.0.0.0/8 are contacted directly as the DIR ECT statement says. PROXY_NAME specifies the server to use for name resolution if local lookup fails. The net 192.168.100.0/24 will be reached via 10.0.4.1. The domain *.private -net.org uses the same socks server. Name resolution will be done on the server . The remaining destination, which is supposed to be the internet, will be redirected via 2 socks servers. The keyword BALANCE is used to change the order in the serverlist randomly. Otherwise the second server will only be used, when the first one can't be contacted. This rule is limited for http requests.

Sample configuration of NEC's socks client:

The syntax of the socks client configuration is similar to the server configuration. For our example the config file /etc/libsocks5.conf will look like:

```
# sample configuration of NEC's socks client
# /etc/libsocks5.conf
#
# 1) All destinations at the private net 10.0.0.0/8 will be reached directly
NOPROXY – 10. –
# 5) Destinations at 192.168.100.0/0 are accessible via Socks -D1
SOCKS5 – 192.168.100. – 10.0.4.1
# 6) Destinations at .private -net.org can be reached via Socks -P1
SOCKS5 – .private-net.org – 10.0.4.1
# 2) The Internet will be reached via the socks gateways Socks -Int1 and Socks-Int2 (http).
# 3) The two internet socks gateways are redundant.
SOCKS5 – – 80 10.0.3.1,10.0.3.2
```

The local class A net is reached directly which signalizes the keyword noproxy. Each server in net 192.168.100.0/24 will be reached by Socks -D1. The same happens to hosts of the domain private-net.org. The remaining request s of typ http will be redirected throgh Socks -Int1. If Socks-Int1 can't be reached it will be tried to send the request to Socks -Int2. To enable remote DNS lookup the environment variable has to be set: SOCKS5_FAKEALLHOSTS. The contrary is SOCKS5_LOCALDNSO NLY. Remote DNS lookup is necessary when you want the domain private-net.org to be resolved on the socks server.

Sample configuration of DANTE's socks client:

The syntax of Dante's client configuration file is easy to read in relation to the previous one .

```
# DANTE's socks client configuration
# /etc/socks.conf
# client doesn't resolve names by itself.
resolveprotocol: fake
# 1) All destinations at the private net 10.0.0.0/8 will be reached directly
route {
from: 10.0.0.0/8 to: 10.0.0.0/8 via: direct
}
# 5) Destinations at 192.168.100.0/0 are accessible via Socks -D1
route {
from: 10.0.0.0/8 to: 192.168.100.0/24 via: 10.0.4.1 port = 1080
}
# 6) Destinations at .private-net.org can be reached via Socks -P1
route {
from: 10.0.0.0/8 to: .private-net.org via: 10.0.4.1 port = 1080
}
# 2) The Internet will be reached via the socks gateways Socks -Int1 and Socks-Int2
(http).
# 3) The two internet socks gateways are redundant.
route {
from: 10.0.0.0/8 to: . port = 80 via: 10.0.3.1 port = 1080
method: username
}
```

First of all resolveprotocol is set to fake. This means the clients don't try to resolve the domain private-net.org but pass it to the socks server. Then all hosts on the local net with the origin 10.0.0.0/8 will be reached directly. All requests to 192.169.100.0/24 will be redirected via Socks -D1. This applies to all hosts that belong to the domain .private -net.org as well. All remaining http requests will be sent to Socks -Int1. Unfortunately you can enter only one socks server and not a list of them. In this case there is no possibility to tell the client about a second server through which the internet can also be reached. A workaround would be to use local name resolution and address the socks server by name. In the dns the socks server will have t wo ip addresses. But on the other hand we then must allow dns lookups of all internet address for the clients.

13 Conclusion

As descried in this paper creating a socksified environment is no magic. When deploying the NEC socks server a lot of requirements can be implemented. Unfortunately it is only restricted to non commercial use. But the Dante socks server meets a wide range of requirements as well. I used it once with openldap user authentication and the Aventail client. Everything worked fine. But I was facing problems when using the Hummingbird client. The Dante developer admited that it is a server side problem but they don't plan to solve this problem in the near future. There are currently no common standards regarding user authentication between s ocks client and server. To avoid problems in this area one should use software from the same provider for both. The other weakness of Dante is proxy chaining. This feature is not yet implemented but will soon be included. But as mentioned in this paper the re are workarounds available.
The presented socks clients work more or less without any problems. There is more effort to be done to configure and to maintain the corresponding config files. One important feature is high availability and load balancing. T he best client for this purpose is the one from Hummingbird. When using the keyword BALANCE the different socks server will be addressed randomly. NEC and Aventail offer a fall back function. When the first server in the list can't be contacted the

next one is tried. This means the fall back server doesn't do anything . A kind of load sharing can be achieved by editing the client's config file. The destinations can be splitted on both socks server with the other one to be the fall back. Dante doesn't offer anything in this field.
As mentioned above, Dante offers a wide range of functionality. But the one who needs more has to apply a commercial socks server. The Permeo socks server is worth looking at.[25]

14 References

[1.] Cheswick, William R.; Bellov in, Steven M., “Firewalls and Internet Security”
Adison Wesley,July 1994

[2.] Hunt, Craig, “TCP/IP Network Administration”
O’Reilly, 1995

[3.] Zwicky, Elisabeth D. ; Cooper Simon; Chapman, D. Brent
“Building Internet Firewalls”
O’Reilly, 2000

[4.] “Dante explanation by Michael Bär”
URL:”http://www.easy-pinguin.de/4/linux/server/dante.html ”

[5.] “Dante’s Homepage”
URL:”http://www.inet.no/dante ”

[6.] “Hummingbirds Socks Homepage”
URL:“http://www.hummingbird.com/products/nc/socks/index.html ”

[7.] “Hummingbirds Socks Homepage: Installation”
URL:“http://www.hummingbird.com/products/nc/socks/install.html ”

[8.] “NEC’s SOCKS Homepage”
URL:”http://www.socks.nec.com/index.html ”

[9.] “NEC’s SOCKS Homepag e: overview of socks server”
URL:”http://www.socks.nec.com/aboutsocks.html ”

[10] “NEC’s SOCKS Homepage: overview of socks version 4”
URL:”http://www.socks.nec.com/socksv4.html

[11.] “NEC’s SOCKS Homepage: use socks5 reference implementation”
URL:”http://www.socks.nec.com/s5use.html

[12.] “NEC’s SOCKS Homepage: configure socks5 reference implem entation”
URL:”http://www.socks.nec.com/s5examples.html

[13.] “NEC’s SOCKS Homepage: advantages of socks server”
URL:”http://www.socks.nec.com /whysocks.html”

[14.] “NEC’s SOCKS Homepage: enable username/password authentication”
NEC “http://www.socks.nec.com/userpass.html ”

[15.] “NEC’s SOCKS Homepage: SocksCap FAQ”
URL:”http://www.socks.nec.com/sockscapfaq.html

[16.] “Socks Proxy Server introduction by Peter Grennan”
URL:” http://www.tldp.org/HOWTO/Firewall -HOWTO-11.html”

[17.] “Firewall introduction by Peter Grennan”
URL:” http://www.tldp.org/HOWTO/Firewall -HOWTO-2.html”

[18.] A SocksV4 Specification:
“SOCKS: A protocol for TCP proxy across firewalls ” by Ying-Da Lee (NEC)
URL:”http://www.socks.nec.com/protocol/socks4.protocol ”

[19.] A brief socks client description:
“Internet connection sharing mini HOWTO: Sharing methods”
URL:”http://www.tele.ucl.ac.be/ELEC2920/1999/Connection/Connection -Sharing-3.html”

[20.] RFC 1928: “The SOCKS Protocol Version 5”

[21.] RFC 1929: “ Username/Password Authen tication for SOCKS V5”

[22.] A free socks client from Hummingbird can be downloaded
URL:”http://www.hummingbird.com/products/evals/nc/socks_form.html ”

[23.] SocksCap Homepage
URL:” http://www.socks.nec.com/reference/sockscap.html ”

[24.] “NEC’s SOCKSv5 Reference Implementation”
URL:” http://www.socks.nec.com/refe rence/socks5.html”

[25.] “permeo Product Overview”
URL:” http://www.permeo.com/Products/ProductOverview.asp ”