MICHAEL SCHRATT

HTTP Tunnel

Difficulty

▨ □ □

Most of all companies only provide a very restrictive environment. While Network and Security Adminstrators do their job, securing the enterprise network from intruders, users are trying to compromise perimeter security to get more than is allowed. Surfing the www and googling provides a huge knowledge on how to greak firewalls, proxies, anti-virus appliances and so on.

Surfing the web is one thing users are allowed to do inside a company. What does it technically mean to surf the web? To access the WWW there must be at least two open ports for allowed outbound connections. Port 80 is used for HTTP and Port 443 is used for HTTPS (see Table 1. for essential port numbers).

It is always easy to create a security branch from inside to outside. Covert Channel Technologies are wide spread and simply every user can make use of it because of easy to understand How-Tos. 100 procent of security can not be achieved, but what you can do is to make it difficult by taking counter measures. According to Covert Channels, if there is any traffic allowed, the protocol available can be used as transport medium and due to this, it is very difficult to detect that traffic.

What I want to demonstrate, is how to hide tracks using HTTPTunneling techniques. I will introduce two user friendly tools and some measures you can consider to prevent tunneling. In our case, traffic looks like normal HTTP/HTTPS Traffic. If there are any anomaly detection systems, it could be that httptunnel traffic produces alert events.

Motivation to use Covert Channels

- Surf on denied websites,
- chatting via ICQ or IRC,
- access private servers in the internet for remote administration,
- downloading files with filtered extensions,
- downloading files with malicious code.

Who can make use of it?

- Hackers,
- disgruntled employees,
- users from the internal network.

Easy to use Tools - GNU ttptunnel

Information extracted from *http://www.nocrew.org/software/httptunnel.html*

`httptunnel` creates a bidirectional virtual data connection tunneled in HTTP requests. The HTTP requests can be sent via an HTTP proxy if so desired.

This can be useful for users behind restrictive firewalls. If WWW access is allowed through HTTP proxy, it is possible to use `httptunnel` and, say, telnet or PPP to connect to a computer outside the firewall. `httptunnel` is written and maintained by Lars Brinkhoff.

Httptunnel is also available as windows binary.

SSH for Windows and Linux

A way to access a shell was former made by the use of telnet.

Telnet is now considered as unsecure due to plaint text transfer. It is possible to sniff telnet traffic on the network to get usernames and

WHAT YOU WILL LEARN...

How to establish HTTP tunneling.

Which tools are in the wild.

What the purpose of tunneling is,

and what possibilities of covert channel techniques there are.

WHAT YOU SHOULD KNOW...

How to use the Linux & Windows operation system.

Tunneling basics.

Knowledge about TCP/IP networks, especially Layer 4 & 5.

How to use a network analysing tool, for example Wireshark, tcpdump.

passwords of different users. On Linux versions after january 2002 you already have OpenSSH installed.

SSH has replaced telnet and has improvments like encrypted traffic. SSH is also called Secure Shell.

Not only encrypted traffic is a reason to use SSH, but also secure file transfer and an enhanced authentication facility. For Windows machines it is possible to get OpenSSH as Windows Binary.

An already wide spread and known SSH client for windows and unix systems is Putty.

Putty is a free available graphic tool which implements telnet and SSH.

Main Problem of Transfer

The most available ports allowed for outbound connections are as mentioned before port 80 for unencrypted HTTP traffic

Table 1. *Essential Port Numbers*

Port Number	Service
20 – 21 / TCP	FTP
22 / TCP	SSH
23 / TCP	Telnet
25 / TCP	SMTP
53 / TCP UDP	DNS
80 / TCP	HTTP
110 / TCP	POP3
143 / TCP UDP	IMAP
161 – 162 / TCP UDP	SNMP
443 / TCP	HTTPS
1080 / TCP	SOCKS Proxy
3128 / TCP	Squid Proxy
5190 / TCP	ICQ – AOL Messenger
6660 – 6669 / TCP	IRC

GNU – What is it?

GNU is an operating system which consists only free software. The GNU Project includes known tools like GCC, binutils, bash, glibc and coreutils. GNU GPL is a licence which can be used for software to mark it as free software. It is called Gerneral Public Licence and has the might to forbid giving any restrictions on programs. Futher information can be found at *http://www.gnu.org*

IANA

See *http://www.iana.org/ for more information.*

Legality and Ramifications

Without addressing every country's laws, there can be sanctions and legal proceedings if using covert channels in corporate networks. Read the companies policies detailed to become familiar with. Be warned and do not use covert channels just for fun. There may be corporate agreements to tunnel data to business partners, for example. This is to ensure that nobody else can listen to your transmission of sensible enterprise information.

Covert Channel Techniques

Covert Channel Hacking is an insider attack to inititate connections from the trusted network to an untrusted network. Different types mentioned below:

Direct Channel Techniques

- ACK Tunnel
- TCP Tunnel (telnet, ssh)
- UDP Tunnel (snmp)
- ICMP Tunnel

Proxified Channel Techniques

- Socks SSL Tunnel
- HTTPS Tunnel
- DNS Tunnel
- FTP Tunnel
- Mail Tunnel

Warning

Using Covert Channels to transfer data out of your companie's network must not be a legal activity (see *Legality and Ramifications*. for more information).

Perimeter Security

Perimeter Security comprises Firewall – Technologies, Packet Filtering, Stateful – Inspection, Application Proxies, *Virtual Private Networks* (VPN), HTTP Proxies, Security Gateways, *Intrusion Detection* (IDS), *Intrusion Prevention* (IPS) up to Bollards, Fencing, Vehicle Barriers, Security Controls (see Figure 1).

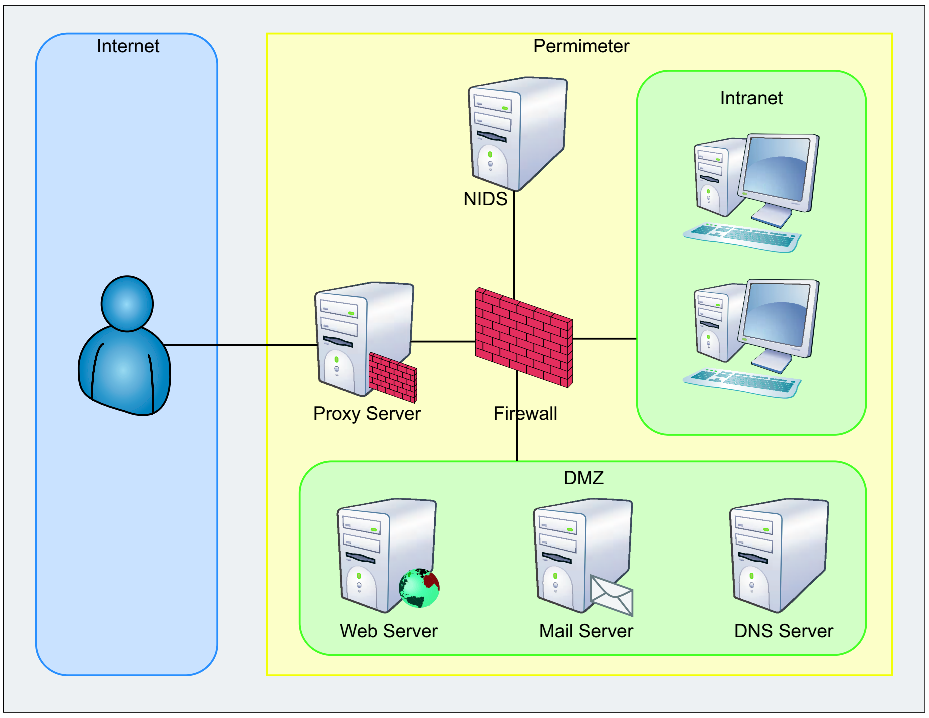


Figure 1. *Network Perimeter Security*

and port 443 for encrypted transfer or HTTPS.

Lets assume, we want to access port 22 for SSH on our server in the internet. Due to firewall restrictions, it is not possible to connect directly on port 22 to open a shell.

Solving the problem with httptunnel

Have a look at figure 5. to see how our tunnel will go through firewalls and proxies. Bypassing content filtering and signature based detection systems due to encryption provided by SSH.

What the main job belongs to is to establish the HTTP tunnel, connect to a shell through the tunnel and what you get is an SSL Traffic based HTTP tunnel with encryption, authentication and integrity.

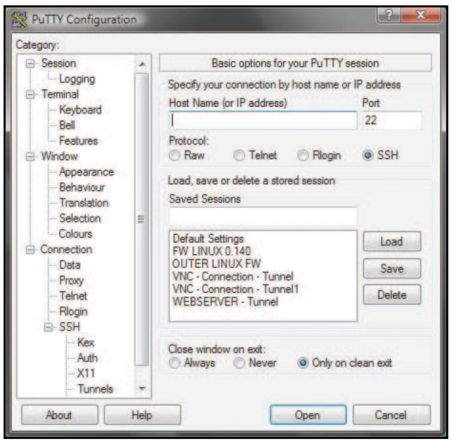
Needed Environment Inside and Outside

Enterprise Side:

- Workstation with internet access, at least one service must be allowed for outbound connections,
- `httptunnel` client,
- ssh client.

Home Side:

- Workstation with internet access,
- `httptunnel` server with correct configuration,
- ssh server daemon with correct configuration (Configuration described in Configure of Services),
- Any service running which you want to access remotely.



Specify your connection by host name or IP address
Host Name (or IP address):
Port: 22
Protocol: Raw Telnet Rlogin [✓ SSH]
Load, save or delete a stored session
Saved Sessions
Default Settings
FW LINUX 0.140
OUTER LINUX FW
VNC - Connection - Tunnel
VNC - Connection - Tunnel1
WEBSERVER - Tunnel
Close window on exit: Always Never [✓ Only on clean exit]

Figure 2. *SSH Client – Putty*

Configure of Services

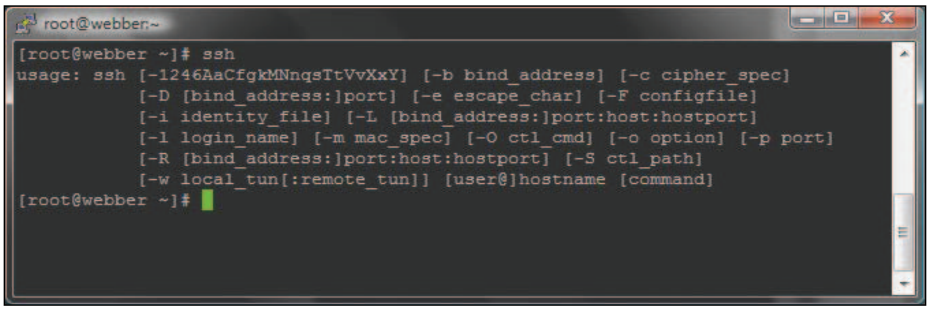
Configure httptunnel Server. Setting up a tunnel is very easy. `Httptunnel` is a command-line tool with several functions.

Belonging to the environment setup described at *Needed Environment Inside and Outside* there are some possibilities that could be used to start and configure `httptunnel`.

Commands:

- `hts -forward-port localhost: 22 443` (tunnel port 443 to 22), or the same
- `hts -F localhost:22 443`

If you do not have root rights you can use unprivileged ports above 1024, for example



```
[root@webber ~]# ssh
usage: ssh [-1246AaCfgkMNnqsTtVvXxY] [-b bind_address] [-c cipher_spec]
           [-D [bind_address:]port] [-e escape_char] [-F configfile]
           [-i identity_file] [-L [bind_address:]port:host:hostport]
           [-l login_name] [-m mac_spec] [-O ctl_cmd] [-o option] [-p port]
           [-R [bind_address:]port:host:hostport] [-S ctl_path]
           [-w local_tun[:remote_tun]] [user@]hostname [command]
[root@webber ~]#
```

Figure 3. *SSH Client – Linux*

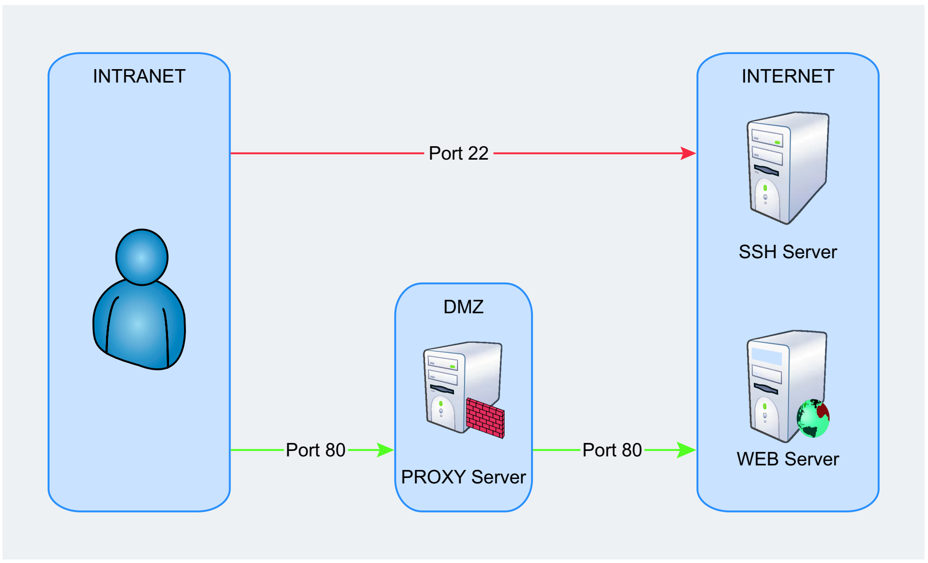


Figure 4. *Transfer Problem*

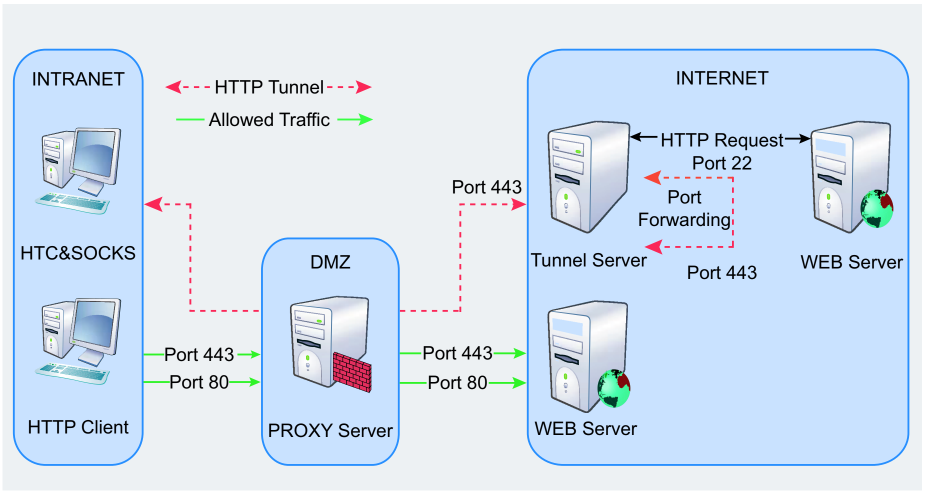


Figure 5. *Solved Transfer Problem*

- `hts -forward-port localhost:22 40000`
- `hts -help`

If our httptunnel server is up and running, it should look like described in Figure 7.

In Addition, our defined port 443 should be LISTENING.

Configure SSH Service

To provide full compatibility with your tunnel make the changes listed in Listing 1.

Final Step: Open Tunnel and connect to the SSH Server

Most work is done, and the final step is to open our tunnel. So, we need to be familiar with the httptunnel client. The simplest way to open a tunnel is:

- `htc --forward-port 10001 192.168.11.240:443`

So, we say, forward local port 10001 to our httptunnel server with ip address 192.168.11.240 on port 443. We are able to prove the established http tunnel by using netstat. Port 10001 has to be in an LISTENING state. If so, start your ssh client and connect to port 10001 on localhost:

- `putty -P 10001 root@localhost` or,
- `ssh -p 10001 root@localhost` or use `-l` for `login _ name` parameter.

See Figure 3. for available SSH parameters.

Enter your credentials if required. From now, you have opened a HTTP Tunnel and connected through it to use the server's shell. In that way, you are only able to use that opened shell to run commands on the server.

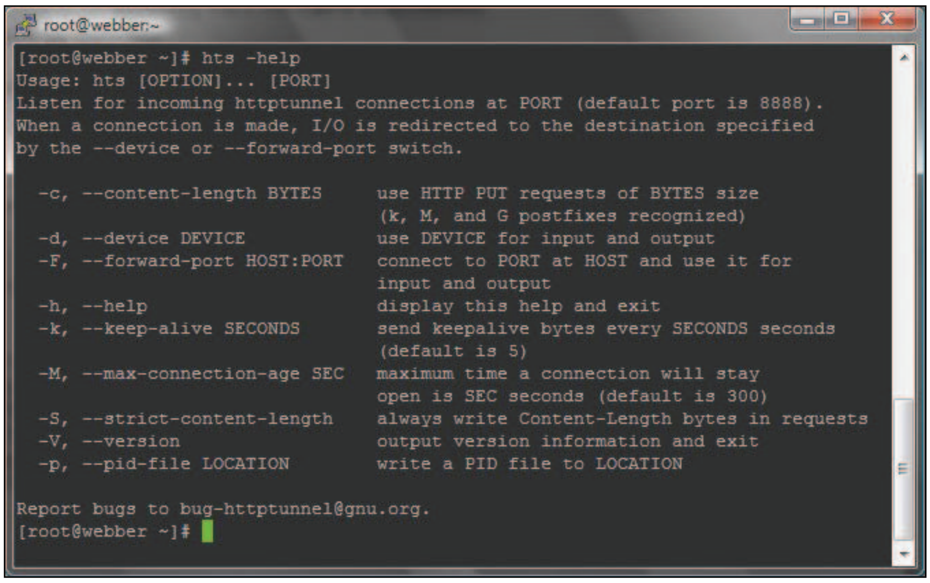
You could use SCP instead, to move data over the tunnel. But that should not be the only thing we want to achieve. Now, we are going to setup a local proxy and use it for other applications like IRC, Skype. Every application that has the ability to use a SOCKS Proxy is welcome.

You are able to use your private email server for sending mails or access your POP, IMAP Server through your tunnel. That is only the question how you make use of port forwarding with your ssh client.

More Practice

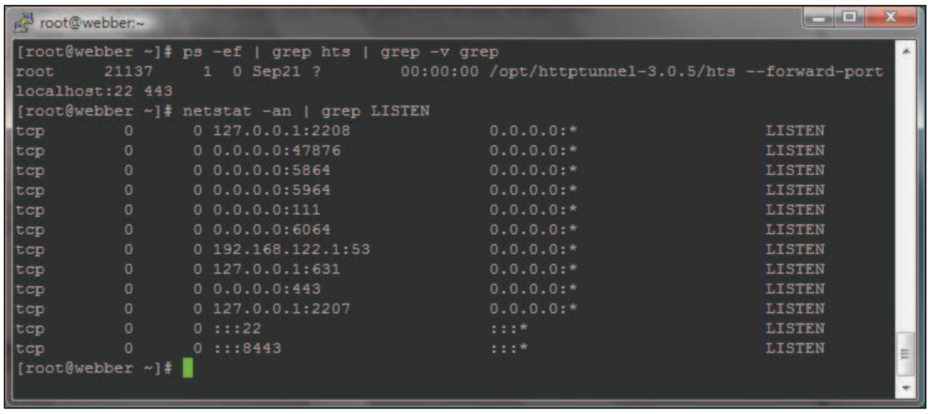
Create your own SOCKS Proxy

- `htc -forward-port 10001 192.168.11.240:443` (open tunnel),
- `putty -D 1080 -P 10001 root@localhost` (connect to shell



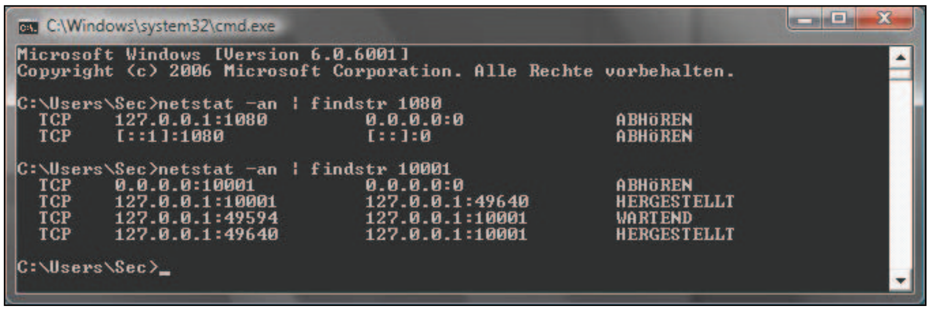
```
[root@webber ~]# hts -help
Usage: hts [OPTION]... [PORT]
Listen for incoming httptunnel connections at PORT (default port is 8888).
When a connection is made, I/O is redirected to the destination specified
by the --device or --forward-port switch.

  -c, --content-length BYTES     use HTTP PUT requests of BYTES size
                                 (k, M, and G postfixes recognized)
  -d, --device DEVICE            use DEVICE for input and output
  -F, --forward-port HOST:PORT   connect to PORT at HOST and use it for
                                 input and output
  -h, --help                     display this help and exit
  -k, --keep-alive SECONDS       send keepalive bytes every SECONDS seconds
                                 (default is 5)
  -M, --max-connection-age SEC   maximum time a connection will stay
                                 open is SEC seconds (default is 300)
  -S, --strict-content-length    always write Content-Length bytes in requests
  -V, --version                  output version information and exit
  -p, --pid-file LOCATION        write a PID file to LOCATION

Report bugs to bug-httptunnel@gnu.org.
[root@webber ~]#
```

Figure 6. *HTS Help Screen*



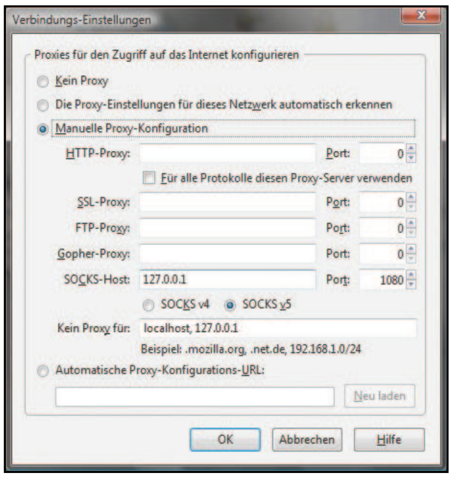
```
[root@webber ~]# ps -ef | grep hts | grep -v grep
root     21137     1  0 Sep21 ?        00:00:00 /opt/httptunnel-3.0.5/hts --forward-port
localhost:22 443
[root@webber ~]# netstat -an | grep LISTEN
tcp        0      0 127.0.0.1:2208              0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:47876               0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:5864                0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:5964                0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:111                 0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:6064                0.0.0.0:*                   LISTEN
tcp        0      0 192.168.122.1:53            0.0.0.0:*                   LISTEN
tcp        0      0 127.0.0.1:631               0.0.0.0:*                   LISTEN
tcp        0      0 0.0.0.0:443                 0.0.0.0:*                   LISTEN
tcp        0      0 127.0.0.1:2207              0.0.0.0:*                   LISTEN
tcp        0      0 :::22                       :::*                        LISTEN
tcp        0      0 :::8443                     :::*                        LISTEN
[root@webber ~]#
```

Figure 7. *HTS Verification*



```
Microsoft Windows [Version 6.0.6001]
Copyright (c) 2006 Microsoft Corporation. Alle Rechte vorbehalten.

C:\Users\Sec>netstat -an | findstr 1080
  TCP    127.0.0.1:1080         0.0.0.0:0              ABHÖREN
  TCP    [::1]:1080             [::]:0                 ABHÖREN

C:\Users\Sec>netstat -an | findstr 10001
  TCP    0.0.0.0:10001          0.0.0.0:0              ABHÖREN
  TCP    127.0.0.1:10001        127.0.0.1:49640        HERGESTELLT
  TCP    127.0.0.1:49594        127.0.0.1:10001        WARTEND
  TCP    127.0.0.1:49640        127.0.0.1:10001        HERGESTELLT

C:\Users\Sec>_
```

Figure 8. *HTC & Proxy Port*


Verbindungs-Einstellungen
Proxies für den Zugriff auf das Internet konfigurieren
OK
Abbrechen
Hilfe
Neu laden

Kein Proxy
Die Proxy-Einstellungen für dieses Netzwerk automatisch erkennen
[✓ Manuelle Proxy-Konfiguration]
HTTP-Proxy: Port: 0
Für alle Protokolle diesen Proxy-Server verwenden
SSL-Proxy: Port: 0
FTP-Proxy: Port: 0
Gopher-Proxy: Port: 0
SOCKS-Host: 127.0.0.1 Port: 1080
SOCKS v4 [✓ SOCKS v5]
Kein Proxy für: localhost, 127.0.0.1
Beispiel: .mozilla.org, .net.de, 192.168.1.0/24
Automatische Proxy-Konfigurations-URL:

Figure 9. *Firefox Proxy Settings*

using local tunnel port and select 1080 as dynamic forwarded port),

- configure your browser like displayed in Figure 9.

I would recommend to use Firefox with any Proxy Management Extension. In that way you are able to quickly switch to other Proxy Settings.

You can use your created SOCKS Proxy with all other apllications that are able to set SOCKS Proxy Settings, for example: Skype, IRC, P2P Software, Browser.

To verify if your SOCKS Proxy works correctly, do the following. Surf the net without proxy and choose Direct Connection in your Proxy Settings of your browser. Go to a website, for example, *http://whatismyip.com* and write down the IP Address printed out.

Next, choose your SOCKS Proxy again, and require your used IP Address again. You will see your IP Address from your own server in the internet. So, your Proxy is working.

You could also use htc (`httptunnel client`) to connect through a proxy and provide credentials for authentication, or define an own User-Agent.

Your are also able to access your internal devices at home. Just type their internal ip address into the address field in your browser.

This has an big advantage, because of just opening one port for incomming connections and using it for your httptunnel server.

Use VNC for remote administration

- Configure VNC Server at your Server outside. Default Ports for VNC are 5900/TCP and 5800/TCP and set your display number. I will use 64 as display number. In that case, the corrected port numbers are 5964/TCP and 5864/TCP,
- `htc -forward-port 10001 192.168.11.240:443`,
- `putty -L 5964:127.0.0.1:5964 -X -P 10001 root@localhost` (-L forward localport 5964 for vnc client, and enable X11 Forwarding with -x),
- Start your VNC Client and connect to `localhost:64 (localhost:<displaynumber>)`.

Use any SMTP Server for mailing

- There must be a SMTP Server running outside,
- `htc -forward-port 10001 192.168.11.240:443`,
- `putty -L 666:<smtpserver>:25 -P 10001 root@localhost`,

Disadvantages of a HTTP tunnel without SSH

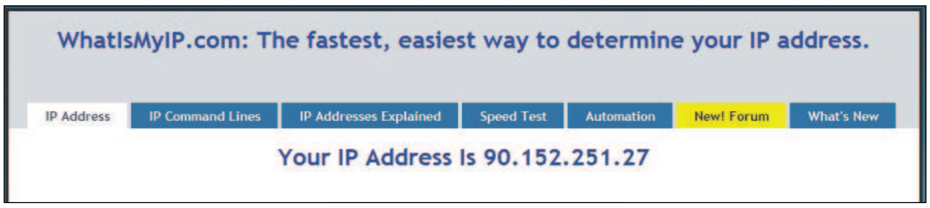
- No encryption, it is possible to sniff your connection,
- No Privacy, anybody can use your tunnel,
- Provides no integrity, your stream could be altered,
- you can only get one established connection through your http tunnel.

Tunnel Security

Provide Integrity, Privacy and Authentication if you use HTTP Tunnel and SSH together.

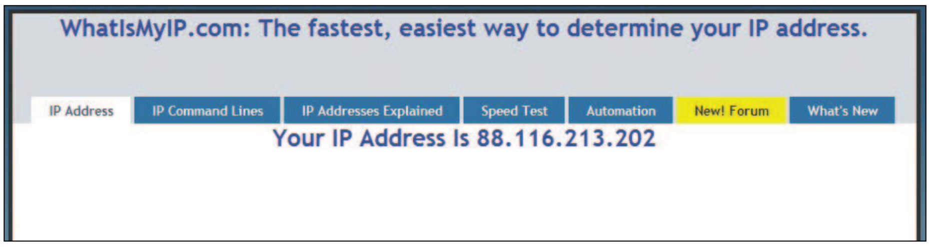
HTTP-CONNECT

The HTTP CONNECT method can be used with a proxy that can dynamically switch to tunnel mode.



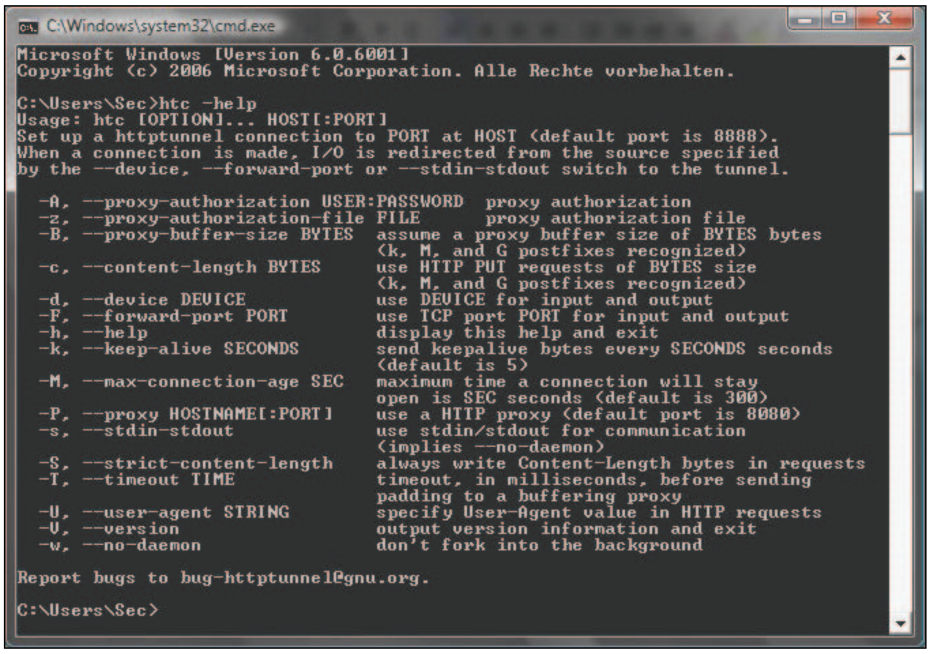
Your IP Address Is 90.152.251.27

Figure 10. *IP Without Proxy – Without Tunnel*



Your IP Address Is 88.116.213.202

Figure 11. *IP with enabled Tunnel*



```
Microsoft Windows [Version 6.0.6001]
Copyright (c) 2006 Microsoft Corporation. Alle Rechte vorbehalten.

C:\Users\Sec>htc -help
Usage: htc [OPTION]... HOST[:PORT]
Set up a httptunnel connection to PORT at HOST (default port is 8888).
When a connection is made, I/O is redirected from the source specified
by the --device, --forward-port or --stdin-stdout switch to the tunnel.

  -A, --proxy-authorization USER:PASSWORD  proxy authorization
  -z, --proxy-authorization-file FILE      proxy authorization file
  -B, --proxy-buffer-size BYTES            assume a proxy buffer size of BYTES bytes
                                           (k, M, and G postfixes recognized)
  -c, --content-length BYTES               use HTTP PUT requests of BYTES size
                                           (k, M, and G postfixes recognized)
  -d, --device DEVICE                      use DEVICE for input and output
  -F, --forward-port PORT                  use TCP port PORT for input and output
  -h, --help                               display this help and exit
  -k, --keep-alive SECONDS                 send keepalive bytes every SECONDS seconds
                                           (default is 5)
  -M, --max-connection-age SEC             maximum time a connection will stay
                                           open is SEC seconds (default is 300)
  -P, --proxy HOSTNAME[:PORT]              use a HTTP proxy (default port is 8080)
  -s, --stdin-stdout                       use stdin/stdout for communication
                                           (implies --no-daemon)
  -S, --strict-content-length              always write Content-Length bytes in requests
  -T, --timeout TIME                       timeout, in milliseconds, before sending
                                           padding to a buffering proxy
  -U, --user-agent STRING                  specify User-Agent value in HTTP requests
  -V, --version                            output version information and exit
  -w, --no-daemon                          don't fork into the background

Report bugs to bug-httptunnel@gnu.org.

C:\Users\Sec>
```

Figure 12. *HTC Help Screen*

On the 'Net

- *http://www.gnu.org/* – GNU Project,
- *http://www.iana.org/* – Internet Assigned Numbers Authority,
- *http://www.iana.org/assignments/port-numbers/* – List of Port Numbers,
- *http://www.nocrew.org/software/httptunnel.html* – httptunnel software,
- *http://www.neophob.com/serendipity/index.php?/archives/85-GNU-HTTPtunnel-v3.3-Windows-Binaries.htmlss.full.link* – httptunnel win32 binaries,
- *http://www.w3.org/Protocols/rfc2616/rfc2616.html* – RFC 2612, Hypertext Transfer Protocol HTTP/1.1,
- *http://multiproxy.org/* – Proxy Lists,
- *http://www.stunnel.org/* – Stunnel,
- *http://www.ethereal.com/* – Ethereal, Wireshark,
- *http://www.snort.org/* – Snort IDS,
- *http://www.openssh.org/* – OpenSSH,
- *http://sshwindows.sourceforge.net/* – OpenSSH for WIndows,
- *http://openvpn.sourceforge.net/* – OpenVPN,
- *http://www.netfilter.org/* – Iptables and Netfilter,
- *http://www.htthost.com/* – TCP/IP through HTTP,
- *http://www.dnstunnel.de/* – DNS Tunneling,
- *http://thomer.com/icmptx/* – ICMP Tunneling,
- *http://www.ntsecurity.nu/toolbox/ackcmd/* – ACK Tunneling.

Listing 1. *SSH Configure*

```
/etc/ssh/sshd_config
AllowTcpForwarding yes
#Specifies whether TCP forwarding is permitted

GatewayPorts yes
#Specifies whether remote hosts are allowed to connect to ports forwarded for the
                       client.

X11Forwarding yes
#The connection to the X11 display is auto-matically forwarded to the remote side in
                       such a way
#that any X11 programs started from the shell (or command) will go through the
                       encrypted
#channel, and the connection to the real X server will be made from the local machine.

PermitTunnel yes
#Support for VPN Tunneling
```

Listing 2. *Sample Firewall Ruleset*

```
# drop suspicious packets and prevent port scans
iptables -A INPUT -p tcp --tcp-flags ALL FIN,URG,PSH -j DROP
iptables -A INPUT -p tcp --tcp-flags ALL ALL -j DROP
iptables -A INPUT -p tcp --tcp-flags ALL SYN,RST,ACK,FIN,URG -j DROP
iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP
iptables -A INPUT -p tcp --tcp-flags SYN,RST SYN,RST -j DROP
iptables -A INPUT -p tcp --tcp-flags SYN,FIN SYN,FIN -j DROP

# A Way to prevent ACK Tunneling, a new connection must be initiated with an SYN Flag
                       ON.
iptables -A INPUT -p tcp ! --syn -m state --state NEW -j DROP
# SYN-Flood Protection
iptables -N syn-flood
iptables -A INPUT -p tcp --syn -j syn-flood
iptables -A syn-flood -m limit --limit 1/s --limit-burst 4 -j RETURN
iptables -A syn-flood -j DROP
# Reject HTTP CONNECT Queries
iptables -I INPUT -p tcp -d 0/0 --dport 80 -m string --string "CONNECT" -j REJECT
# Limit Connections
iptables -p tcp -m iplimit --iplimit-above 2 -j REJECT --reject-with tcp-res
```

- configure your mail client to use `localhost:666` as Outgoing Mailserver.

Counteractive Measures

- Disallow unimportant traffic (Listing 2.),
- close unneeded ports and stop unnecessary services,
- use Stateful Inspections to prevent ACK Tunneling,
- set timeouts for connections to prevent Covert Timing Channels,
- use Content Filtering,
- use HIDS and NIDS,
- use Proxies with Authentication,
- disallow HTTP-CONNECT Queries,
- make use of Anti Virus Software and Anti Spyware Software,
- inspect logfiles an a regularly basis,
- have a detailed look at suspicious traffic,
- monitor your network and build statistics of traffic.

Conclusion

You see, building up a tunnel is not very difficult. You only need little experience and understanding. httptunnel is also a recommended tool in penetration testing. You can hide your tracks to ensure not to be protected by any perimeter security devices. Altough, there are some methods of anomaly detection measures, for example, to compare incomming http traffic to outgoing. A security baseline would be that incomming http traffic is likely to be higher than outgoing. If you have got that specific anomaly, this could be hidden traffic. Also the encryption of the SSL Tunnel exhibits barriers in detecting hidden traffic.

There are countries where it is not allowed to use encryption. And once again, you can implement all measures for making it difficult to attack, but there may be further security branches due to wrong configurations, unknown signatures, covert channels, user ignorance and so forth. Finally,

I ask you, not to use above mentioned techniques for illegal matters. Before making use of it, get familier with provisions of the countrie's law.

Michael Schratt
Michael Schratt deals with Network & Operational Security, is an enthusiastic programmer and has big skills in WebApplication Security. His basic job is to maintain enterprise monitoring systems and endpoint security on unix and windows machines.
Contact: *mail@securityinside.info*