

stunnel – secure “tunnel” between TCP applications

David Morgan

© David Morgan 2006

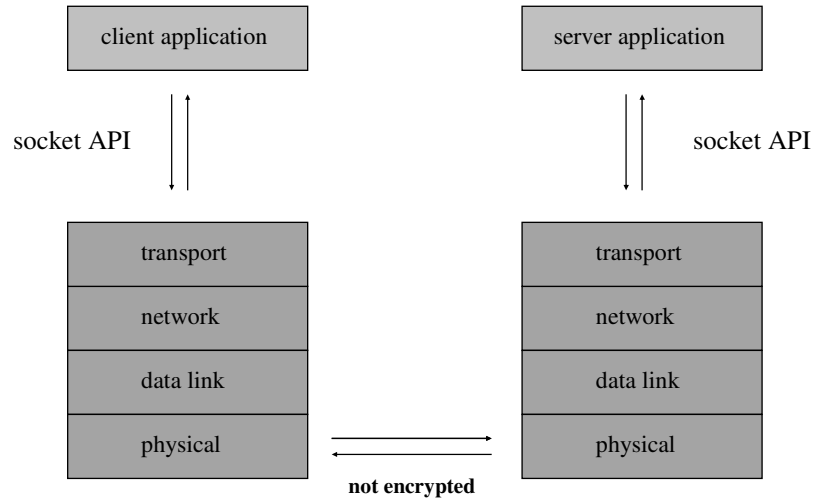
Encrypt the talk between clients and servers who don't

“The stunnel program is designed to work as SSL encryption wrapper between remote clients and local (inetd-startable) or remote servers. The concept is that having non-SSL aware daemons running on your system you can easily set them up to communicate with clients over secure SSL channels.

stunnel man page

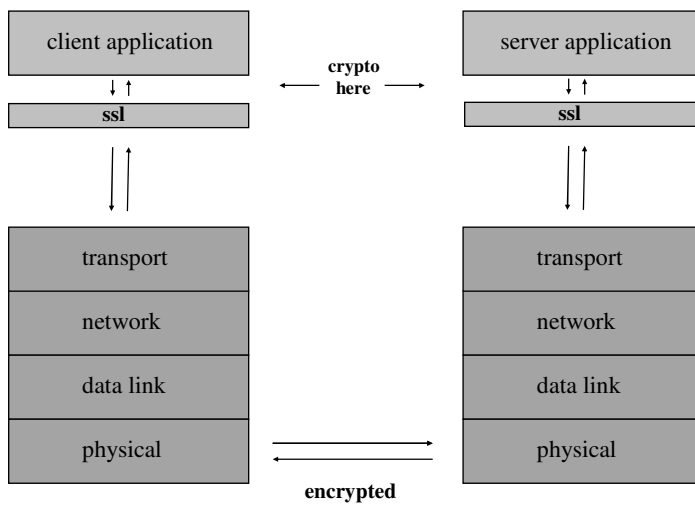
© David Morgan 2006

Ordinary ssl-unaware applications



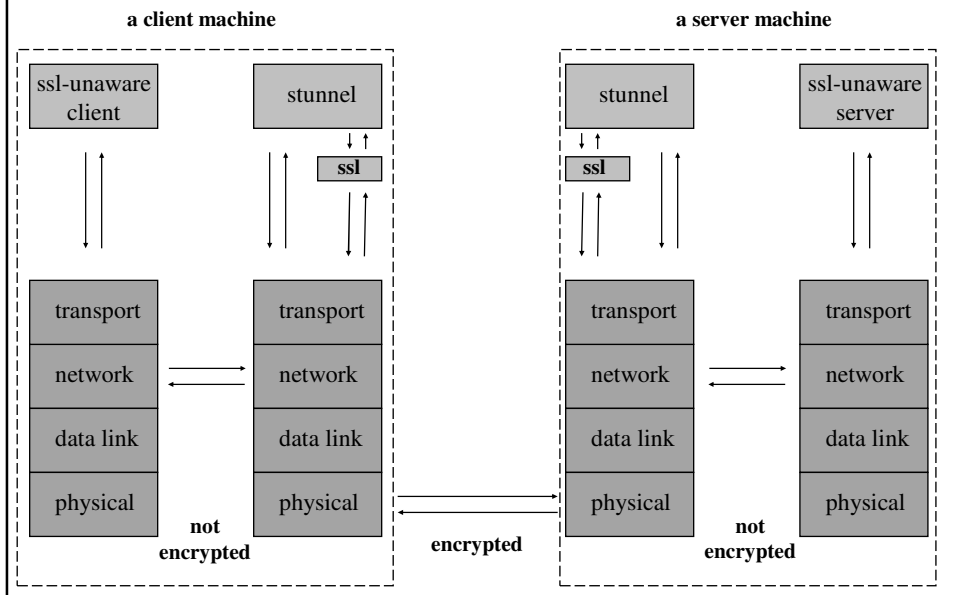
© David Morgan 2006

SSL-aware applications

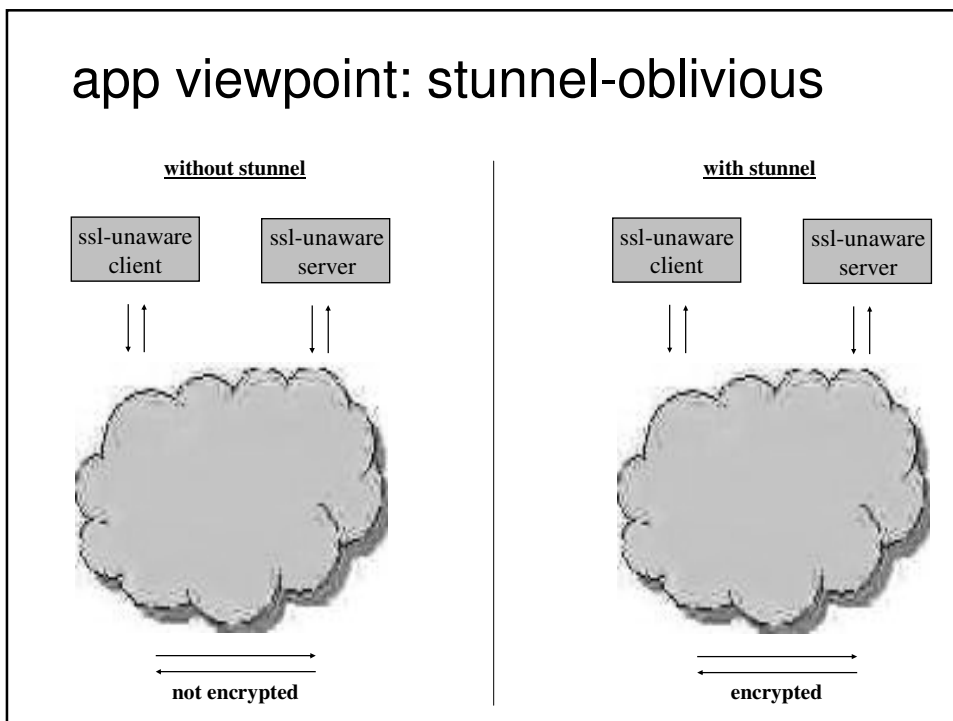


© David Morgan 2006

stunnel – 3 TCP conversations



app viewpoint: stunnel-oblivious



Ports: non-stunnel scenario

client application

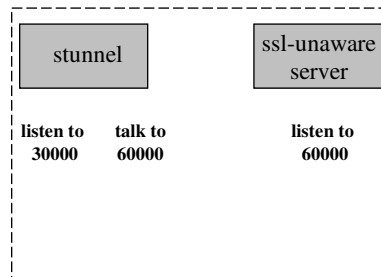
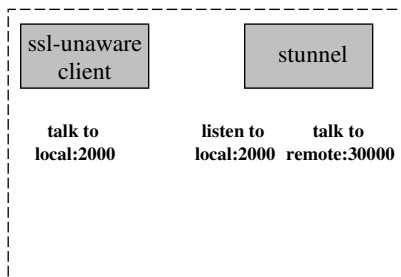
talk to
remote:60000

server application

listen to
60000

© David Morgan 2006

stunnel – 3 TCP conversations



Vanilla config files

```
# stunnel client
```

```
client=yes
```

```
[stunnel service name]
```

```
accept = 127.0.0.1:2000
```

```
connect = 192.168.3.12:30000
```

```
# stunnel server at 192.168.3.12
```

```
cert = /etc/stunnel/stunnel.pem
```

```
[example service name]
```

```
accept = 30000
```

```
connect = 60000
```

© David Morgan 2006

stunnel server needs certificate

- create it with

```
cd /etc/stunnel
```

```
openssl req -new -x509 -days 3650 -nodes -out stunnel.pem -keyout stunnel.pem
```

- reference it in stunnel server's config file
- <http://www.stunnel.org/faq/certs.html#ToC5>

© David Morgan 2006

With and without

```
root@monarch:/etc/stunnel - Shell - Konsole
Session Edit View Bookmarks Settings Help

[root@monarch stunnel]# cat client3-direct-to-server.c | grep -1 "inet_addr"
address.sin_family=AF_INET;
address.sin_addr.s_addr = inet_addr("192.168.3.12");
address.sin_port = htons(60000);
[root@monarch stunnel]#
[root@monarch stunnel]# ./client3-direct-to-server
char from server = B
[root@monarch stunnel]#
[root@monarch stunnel]# cat client3-thru-stunnel.c | grep -1 "inet_addr"
address.sin_family=AF_INET;
address.sin_addr.s_addr = inet_addr("127.0.0.1");
address.sin_port = htons(2000);
[root@monarch stunnel]#
[root@monarch stunnel]# ./client3-thru-stunnel
char from server = B
[root@monarch stunnel]#
```

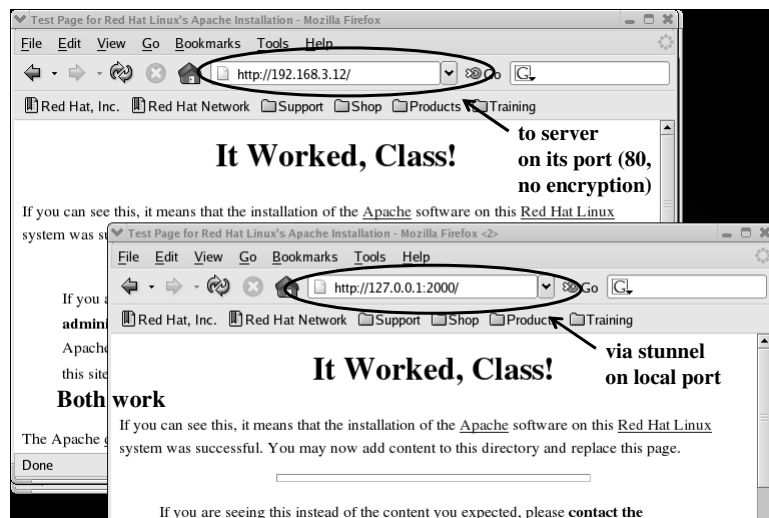
Annotations in the terminal window:

- Both work**: points to the execution of both client programs.
- via stunnel on local port**: points to the local IP address "127.0.0.1" in the second client's configuration.
- to server on its port (no encryption)**: points to the remote IP address "192.168.3.12" in the first client's configuration.

see concurrent capture files

© David Morgan 2006

With and without



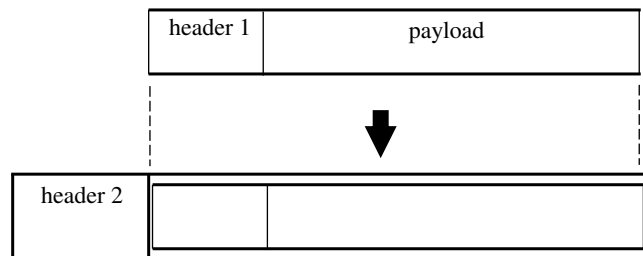
© David Morgan 2006

stunnel's not really a tunnel

- stunnel is a conversation endpoint
- and a (different) conversation startpoint
- arriving packets are stripped of header at endpoint
- their content repackaged, new header, at startpoint
- headers do not nest/accumulate as in tunnels

© David Morgan 2006

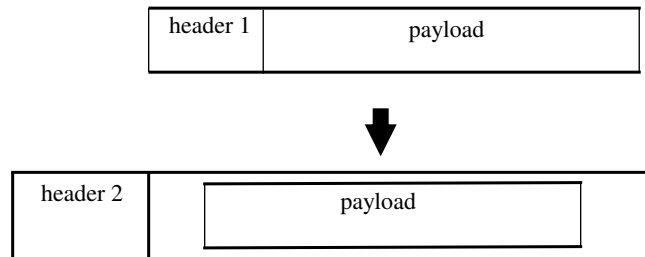
True tunneling



headers accumulate

© David Morgan 2006

Payload forward/relay/proxy



headers replace each other

© David Morgan 2006

info

- <http://www.stunnel.org/>
- <http://freshmeat.net/articles/view/1781>

© David Morgan 2006