November 13th - 2005
Written by rootn0de

Blocking Skype
Using Squid and OpenBSD

Abstract :

After much digging online for an effective way to stop this pesky app that is highly de-centralised and a big pain to blocked, I finally found a way to do quite nicely. It has been working perfectly fine on our corporate network, and we have had no complaints of users being denied access to legitimate web destinations (that are in compliance with our security policy of course). I used Squid-proxy running on an OpenBSD server to carry out the below. The choice of OS to run the proxy on is subjective (I chose OpenBSD as my network OS of choice for its proven security record and excellent reliability) and has no effect over the actual blocking mechanism. The same can be accomplished on any other BSD or Linux flavour.

Background :

This basic write-up will not delve deeply into the operation of Skype, but will quickly highlight the main challenges of blocking this application. As mentioned, the below is not an accurate study of how Skype operates, and is not be a comprehensive analysis of its behaviour :

1) Skype will initially attempt to contact supernodes, the IPs of which are in a file stored along with the other files that Skype installs. The first method of contact is direct. The source ports that Skype attempts to connect from are non-default ports. From my observations I could see that the UDP source port 1247 is the initial control channel. Once the connection is established, the rest of the

communications is done in TCP over non-default source ports with ranges sweeping from 2940-3000.

In general, any company that is serious about its security policy would have strict egress filtering rules, which makes identifying the non-default source/destination ports that Skype uses irrelevant since they would be blocked anyway.

2) If the above fails, Skype will use the proxy server specified in Internet Explorer, and attempt to tunnel the traffic over port 443 using the SSL protocol. The destination IPs are of course random as above, which makes destination blocking out of the question. The only option left is to block SSL, which is not really a solution, unless you want to end up excluding all legal SSL destinations.

Deleting the user's proxy settings would also disallow Skype from connecting. That would however leave the user without internet access. Even if the user had no proxy settings, and the proxying was done transparently (which would definitely include proxying http and https traffic), the Skype traffic (SSL) would again be transparently proxied, which puts us back at square one.

The Alternative That Works :

Internet access services in our corporate workplace are provided by our proxy servers. The setup is basically Squid-proxy running over OpenBSD. PF (packet filter, OpenBSD's built-in firewall) takes care of all the egress/ingress filtering, and the rest of the content filtering is done in Squid using custom-written access-lists.

Blocking Skype's default operation was a no-brainer, as our strict egress filtering rules block all outgoing traffic. The problem was with Skype detecting the user's proxy server, and tunneling its traffic over Squid. Upon checking Squid's access logs, all we could see was requests made by the user's machines using the 'Connect' method to random destination IPs. As mentioned above, blocking SSL or the 'Connect' method, means blocking access to all legitimate websites that use SSL (Hotmail, Yahoo, E-banking, E-commerce websites, e.g any website that is secured by SSL). Should you go down that road, you would have to explicitly allow all permitted destinations (an ongoing technical nightmare).

The catch in successfully blocking Skype given all of the above, would be to block access to requests made by clients, to destination specified by their numeric IP address, AND using the 'Connect' method to tunnel the Skype data. I have done that simply by writing an access list in Squid that achieves just that.

The access-list is in regex (regular expression) format that identifies numeric IP addresses. The access-list further specifies the connection method that the client is using. In Squid the 'Connect' method is conveniently called 'Connect' as well. The access list then is of the following form :

```
# Your acl definitions

acl numeric_IPs urlpath_regex ^[0-9]+\.[0-9]+\.[0-9]+\.[0-9]+
acl connect method CONNECT



# Apply your acls

http_access deny connect numeric_IPs all
```

I have had no problems at all with the above setup, and as a result, only Skype is blocked, as most(if not all) companies serious about having a web presence have registered domains and hence are referenced by their FQDN URLs.

The blocking was so effective, that one user told me days later, he had not bothered calling me the first 2 days to complain about lost SKype connectivity because he thought there was an actual problem with Skype. Apparently, Skype is so good at getting around firewalls. On the third day however, the user called to inquire whether Skype had effectively been blocked. The answer was a definitive yes :)

Should you find the above useful, I would appreciate all comments emailed to vi_cipher@yahoo.com. Enjoy !