Making Skype work properly with HTTPS inspection enabled *incorporating* ‘To probe bypass or not to probe bypass’

Original: 16th May 2018

Updated: 2nd July 2018

Updated: 14th September 2018

Updated 1st October 2018 (Thanks go to Marcel Wildenberg for pointing out the error in the HTTPS policy example)

Author: John Fenoughty

E&OE – I have checked and double checked for typing or logical errors but there may be the odd thing, please feel free to let me know of any additions or changes that might make a material difference .

Contents

Introduction

This document was originally about HTTPS inspection in general, and in particular about the probe bypass feature, with Skype being an example (along with Dropbox and one or two other sites that we were testing) but the testing of all sites, once other extraneous factors were removed showed that turning off the probe bypass is the best move we could possibly make. I know this is a conclusion but the rest of the document discusses this further…

I thought it was going to be a case of balancing one need against another or about certain sites working and others not in each state (PB on or off) but our findings revealed nothing of the sort. There are some sites/apps which have their own very special situations (Skype being notable among these) but the disabling of the Probe Bypass (which I have up to now been running with it enabled on most sites) has proved to be a *huge* improvement. It is important to note, that this needs to be done with care and with time to address issues importantly because:

It changes a fundamental part of the SSL handshake inspection and where sites were previously not being detected and categorised, they may now be correctly inspected and categorised; thus applying parts of the Application & URL filtering policy which may not have previously worked for certain sites (those using SNI primarily). The effect can be that sites which previously were accessible to users now (having disabled PB) are not. This could lead us to jump to the conclusion that turning of the PB actually broke them – whereas to the contrary it enabled them to be inspected and seen and they were against policy (either intentionally or unintentionally) all along! This needs managing at the time of changing the state of the PB.

In the major example case of Skype where it had previously been 'just about' working, this process once initiated will be very likely to break it, *however* following through this entire document should leave Skype specifically in the best state since enabling HTTPS inspection on a network.

Additional – SK104717 (HTTPS Inspection Enhancements in R77.30 and above) still states 'Non-Browser Applications connections are dropped when HTTPS Inspection is enabled (even if bypass is configured).' – while this will no doubt be true for ***some*** non-browser applications testing shows that many important ones can now be made to work, Skype, Dropbox, Box, Webex to name a few. If an organisation finds a specific one which just can't be made to work then hopefully it will be sufficiently small a requirmnt that a dedicate computer without HTTPS inspection can be used. If not – then gateway HTTPS inspection may not be appropriate on that site.

Summary

In ***sk114419 - Unable to bypass Skype in HTTPS Inspection policy using a "Category", or a custom "Application/Site" in the rule***

Check Point are telling us not to use HTTPS inspection on 'Skype Clients' - oh dear! Skype or Skype for business is an essential business tool for many organisations and 'Skype clients' could be any workstation in the organisation. We clearly want to use HTTPS inspection on typical workstations, so this approach is, to be quite frank, absolutely no use at all!

I'm pleased to say that, following the below, I now have HTTPS inspection and Skype (both consumer and business versions) working well on multiple sites. It's early days and of course the lovely people at Microsoft can change things on us at any time so we do need to re-test regularly (I think once every two months or so), especially if users report degradation in quality or slow connections etc.

The process for getting Skype to work

Importantly (or vitally IMO) this was all done without using any IP addresses or ranges of IP addresses. We as firewall administrators cannot be attempting to define and maintain IP range objects for a third party hosted application, hosted in a content delivery network of a 4th party; that way madness lies!

Application & URL policy

First we must allow these pre-configured applications in the Application & URL policy – assuming something in the policy blocks any of the tags or categories associated with Skype. These differ for Skype, which to Microsoft and I suppose therefore to the rest of us, means the consumer or non-business version of Skype and Skype for Business (previously known as Lync) is the commercial version; they have a lot in common and a lot of differences too. Other than for the Application & URL filtering element of making this work, I have approached them together rather than separately because there is so much cross-over and any one organisation may use both, or one today and another tomorrow.

The relevant tags & categories are:

Skype (consumer)

Encrypts communications, High Bandwidth, Video Conferencing , VoIP

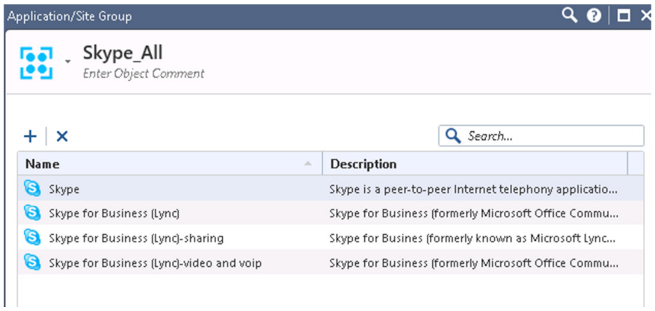
Skype for Business (Lync)

Skype for Business (Lync): Microsoft & Office365 Services, Cloud Services

Skype for Business (Lync)-sharing: Microsoft & Office365 Services, Cloud Services

Skype for Business(Lync)-video and voip: VoIP , Microsoft & Office365 Services, Cloud Services

As a passing note: Of these, I generally avoid using cloud services, Encrypts communications or High Bandwidth in any blocking rules anyway because they are too broadly defined and anything risky in these definitions is (I believe) blockable by other more accurate and better defined categories or tags…



Skype_All

Name	Description
Skype	Skype is a peer-to-peer Internet telephony applicatio...
Skype for Business (Lync)	Skype for Business (formerly Microsoft Office Commu...
Skype for Business (Lync)-sharing	Skype for Busines (formerly known as Microsoft Lync...
Skype for Business (Lync)-video and voip	Skype for Business (formerly Microsoft Office Commu...

A Custom Application with URLs

This was the more difficult bit: we have a custom application/site with the following URLs (not defined as regular expressions) which we put in an **HTTPS inspection bypass rule. This was the task which took the most time; we had to use a ‘quiet’ PC and keep running Skype, testing each feature and noting the various URLs it used!**

The various Skype features require different URL resources and in some cases different protocols.

The features which we tested and succeeded with were:

- Voice Call
- Video Call
- Text Chat (and the notification of a user is typing)
- File transfer
- Screen Sharing (only available between matched editions of ‘Skype for business’ or ‘Skype for non-commercial use’

A Slightly Off-topic Breakout:

This is not essential to the process but was an interesting (and important) lesson learned during this process.

There is an issue here, which we tripped over by meticulous testing. When you add a bypass rule, be sure to limit the source to *ONLY* those devices on the network for which you have already enabled HTTPS inspection. It is common to think, that as this is just an override you can apply it to *any* source and *any* destination but if you do, then you will in effect (partially at least) turn on HTTPS inspection for *ANY* device on the LAN. This can clearly cause very undesirable effects!

This is due to the same underlying reason for the problem of ‘HTTPS Bypass (with Site Category) not working for Servers with Self-Signed Certificate’ discussed in sk11467; this being that if you use the site/category field in a rule, and that rule applies to a connection due to the source and destination matching (if the source and destination are ‘any’ then this is always) then the connection will need to be inspected to establish if the category, URL or application matches.

So, why does this matter? For computers where we did not intend to enable HTTPS inspection, it would then be enabled but only in order to assess if the category in this bypass rule applied, once this is assessed as ‘not the case’ for a given connection then that connection would *not* be inspected. The net result of this is that we don’t see inspection logged for the machines not in the full inspection groups but we *DO* see it break connections to sites that require SNI extension in the SSL "Client hello" packet – this represents a significant section of the Internet. The example for my testing was https://secure2.sophos.com/en-us/login.aspx which is simply the login page for Sophos’ licence and downloads site; this uses ‘octacdn.com’ for authentication and this in turn uses the SNI extension for the certificate. When HTTPS inspection is (accidentally in this case) enabled for a workstation the symptom we see is that access to these SNI reliant sites eventually fail with ‘This site can’t be reached’ or ‘The connection was reset’. This symptom is of course only the case if the probe bypass mechanism is enabled. For more on this, see the next break-out box.

The symptom of unintended HTTPS inspection caused by an overly-widely defined bypass rule is universal and not *dependant* on the probe bypass being enabled or disabled (while of course PB can affect these resulting situation for the computers and users)

I am told to expect progress in the HTTPS inspection policy in R80.20, I look forward to that!

SEPTEMBER 2018 ADDENDUM: I have observed some unreliability when using an AD user group based User Access ROLE to enforce or bypass HTTPS inspection. It manifests as a site sometimes being inspected and sometimes not – the certificate in the browser clearly shows when it is inspected and when it is not. While HTTPS inspection remains in the R77.30 dashboard rather than fully integrated with the unified policy I am avoiding using these where I could instead define a simple group of computers or a subnet object instead. Where users are essential I am considering using an LDAP group which is more aligned with the older policy methods. This is

To probe bypass or not to probe bypass

Before doing the **URL based exceptions** we need for Skype, we must address the '**to probe bypass or not to probe bypass' conversation.** Check Point explains the probe bypass addition (available but not enabled by default from R77.30 onwards as documented in sk104717) as:

Bypass mechanism was improved to better reflect policy and resolve the above limitations:

- Stop the inspection of the first connection to bypassed sites.
- Allow bypass of Non-Browser Applications connections. **(rather important!)**
- Allow Bypass of connections to servers that require client certificate.
- New probing mechanism eliminates the need to inspect the first connection to an IP address unless it is required by the policy.

But mentions that the limitation: **HTTPS Inspection will not work for sites that require SNI extension in the SSL "Client hello" packet.**

This appears to be a fairly small thing in the SK article, it has no major emphasis and no further expansion on the point is given, however:

Testing shows very clearly that this is no small thing, a huge (and increasing all the time) number of sites that are hosted by cloud service providers or use shared content delivery or authentication systems or are hidden behind security networks such as CloudFlare (limited in this example to those using the (free) 'universal SSL' service provided by CloudFlare) which use the SNI extension to direct connections to the appropriate site on their servers. There are plenty of other sites now using this method of SSL hosting, content delivery networks and federated authentication systems being very typical examples of these.

Vitally for what we are trying to achieve in this case, various Skype URLs are content delivery networks and/or on shared platforms using the SNI technique. So in short, we can't make Skype work (reliably) until we turn off Probe Bypass.

A little aside about SNI: The SNI method has been proved to be vulnerable to a hijacking attack (as Let's Encrypt discovered and they have disabled it as an option in most cases due to this)

Apparently Check Point (according to Amos Reiss) have a fix coming in July for this issue (probe bypass with SNI) on top of R80.10. We'd need to open a call to get the patch, who knows when it will come to a jumbo hotfix in GA.

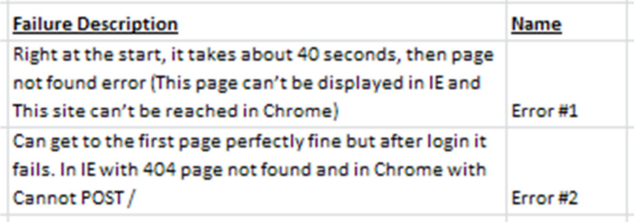
They are going to move the rulebase decision to a later point in the connection when we are able to send the SNI and then cache would be including "verified" SNI. This should allow us to work with SNI in a secure manner. I will need a lot of convincing before I turn probe bypass back on anywhere that matters though!

And never forget: Probe Bypass should not be used if there is a proxy between the Security Gateway and the Internet. But then if you are using a proxy then you already have a lot of problems using the Internet anyway, right?

Some example websites tested before and after disabling Probe Bypass

The following is a little example of some websites which were tested for a particular end user site, showing the 'before and after' results of disabling Probe Bypass. We were checking to see if the theory was good…

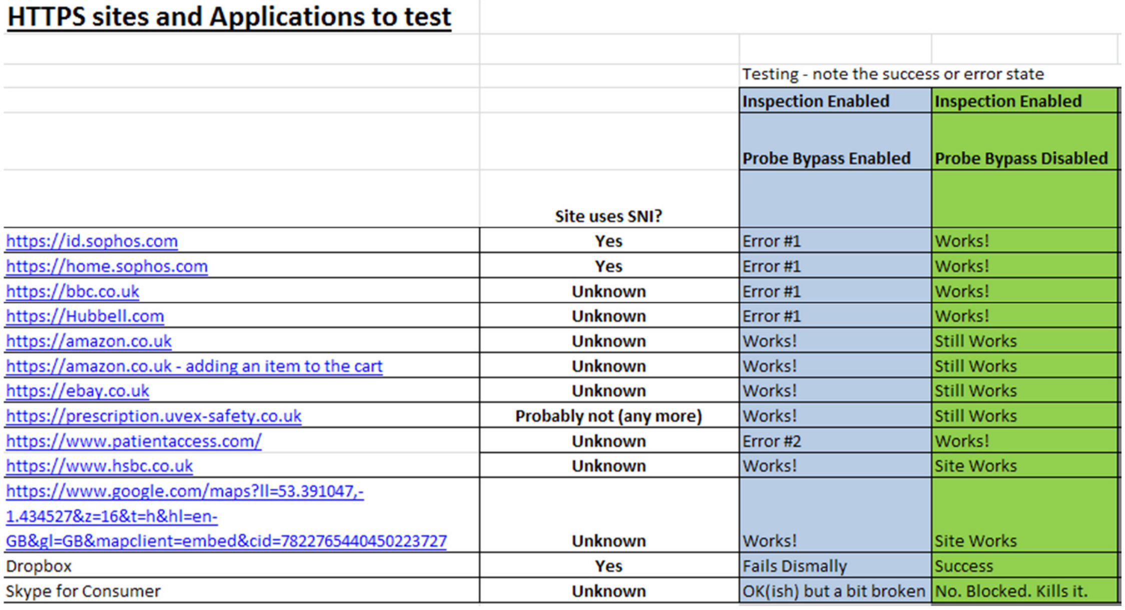
There were two ways that these websites or apps would fail, before or after login. Failure codes were established for these tests were for these two situations:



Failure Description	Name
Right at the start, it takes about 40 seconds, then page not found error (This page can't be displayed in IE and This site can't be reached in Chrome)	Error #1
Can get to the first page perfectly fine but after login it fails. In IE with 404 page not found and in Chrome with Cannot POST /	Error #2

The results were as follows, the blue was before PB disabled the green after. The positivity of these results is very clear!

September 2018 addition – I have revisited these tests since the original time of writing and some of these sites have changed their hosting and even authentication methods; the Internet never stands still!



HTTPS sites and Applications to test

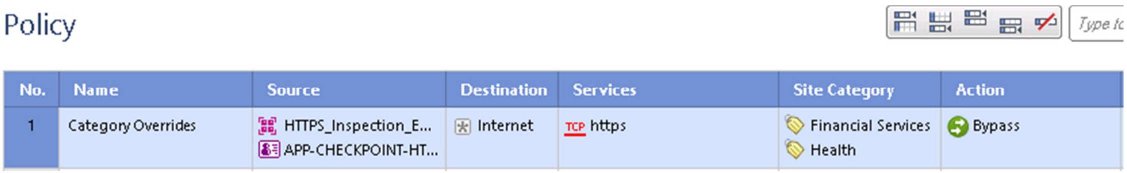
		Testing - note the success or error state	
	Site uses SNI?	Inspection Enabled Probe Bypass Enabled	Inspection Enabled Probe Bypass Disabled
https://id.sophos.com	Yes	Error #1	Works!
https://home.sophos.com	Yes	Error #1	Works!
https://bbc.co.uk	Unknown	Error #1	Works!
https://Hubbell.com	Unknown	Error #1	Works!
https://amazon.co.uk	Unknown	Works!	Still Works
https://amazon.co.uk - adding an item to the cart	Unknown	Works!	Still Works
https://ebay.co.uk	Unknown	Works!	Still Works
https://prescription.uvex-safety.co.uk	Probably not (any more)	Works!	Still Works
https://www.patientaccess.com/	Unknown	Error #2	Works!
https://www.hsbc.co.uk	Unknown	Works!	Site Works
https://www.google.com/maps?ll=53.391047,-1.434527&z=16&t=h&hl=en-GB&gl=GB&mapclient=embed&cid=7822765440450223727	Unknown	Works!	Site Works
Dropbox	Yes	Fails Dismally	Success
Skype for Consumer	Unknown	OK(ish) but a bit broken	No. Blocked. Kills it.

So as we can clearly see, this particular list (which was a moment in time, these specific sites can all change how they work without notice of course), tests elsewhere included all manner of different sites to this particular little subset. The interesting point being the ratio of those which fail with PB on and off; turning it off for these examples left us with 6 of 13 which worked still working, made 6 of 13 which were broken work correctly and as the rest of this document shows we were able to then get the 13th (Skype) to work perfectly as well!

I have been working with various end users on this over the past couple of months and the effect of disabling probe bypass has been ***universally*** positive! The summary of all of my work in this area is: **Disable Probe bypass, have confidence that you will now be able to fix the resulting issues shown in policies with URL overrides in almost all cases and HTTPS inspection as well as use satisfaction will be greatly improved.**

Following the above, the concern about inspecting (for example) financial or health related sites is, (in my opinion) perfectly well addressed by Check Point as follows:

1. Be sure to be on the latest jumbo hotfix, this is really important
2. Enable HTTPS Inspection for the appropriate group of **workstations** (preferable to users)
3. Disable probe bypass – SNI based sites will now work properly! **Also *actual* bypasses (for SNI based sites) in the HTTPS inspection rule-base will now work properly too!**
4. Put in a category based override like this:



Policy

No.	Name	Source	Destination	Services	Site Category	Action
1	Category Overrides	HTTPS_Inspection_E... APP-CHECKPOINT-HT...	Internet	TCP https	Financial Services Health	Bypass

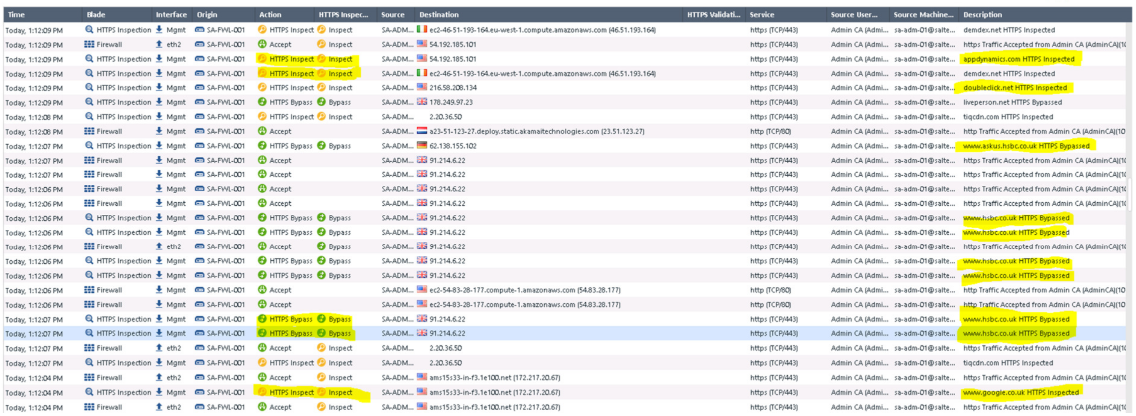
The effect of this will be that the initial connection will not be broken by the Probe Bypass, but *will* be decrypted and **the URL 'resource' will be extracted.**

It will then be checked against the policy and the categorisation of health or financial services (or whatever a given organisation is concerned about not inspecting) will be recognised as applying to that connection and **the rest of the communication from this point** will *not* be inspected!

In any sensible national administration, this should be considered perfectly acceptable. In the UK we don't have any usable case-law about this specific issue yet; however, I for one would be very confident to stand up and defend a policy where we decrypt and inspect the initial connection to discover where a user is browsing to, and then as soon as we are confident that it is a site where we do not wish to be monitoring the user's activity any further we do not do so but we simply allow it to pass fully encrypted.

It is worth observing that the process will recur in the event of an additional or new connection being caused to occur by the cloud application or site to a sub-site or content delivery network but again, only the initial connection will be inspected and then the rest of the connection will continue without the firewalls decrypting it.

In the logs below you can see (reading from the bottom up) www.google.co.uk is inspected, but then www.hsbc.co.uk is bypassed. This is being done by the category based rule above, no custom applications or URLs have been used. Probe bypass is *disabled* and as you can see the website www.askus.hsbc.co.uk is also bypassed showing that all correctly categorised sites and pages will remain encrypted and the user can use his or her banking with confidence that the company system administrator is not 'spying' on his or her finances and even chat sessions with the HSBC online staff.



Time	Blade	Interface	Origin	Action	HTTPS Inspec...	Source	Destination	HTTPS Validati...	Service	Source User...	Source Machine...	Description
Today, 1:12:09 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Inspect	Inspect	SA-ADM...	ec2-46-51-193-164.eu-west-1.compute.amazonaws.com (46.51.193.164)		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	demdex.net HTTPS Inspected
Today, 1:12:09 PM	Firewall	eth2	SA-FWL-001	Accept	Inspect	SA-ADM...	54.192.185.101		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	https Traffic Accepted from Admin CA (AdminCA)(1
Today, 1:12:09 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Inspect	Inspect	SA-ADM...	54.192.185.101		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	appdynamics.com HTTPS Inspected
Today, 1:12:09 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Inspect	Inspect	SA-ADM...	ec2-46-51-193-164.eu-west-1.compute.amazonaws.com (46.51.193.164)		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	demdex.net HTTPS Inspected
Today, 1:12:09 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Inspect	Inspect	SA-ADM...	216.58.208.134		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	doubleclick.net HTTPS Inspected
Today, 1:12:09 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Bypass	Bypass	SA-ADM...	178.249.97.23		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	liveperson.net HTTPS Bypassed
Today, 1:12:08 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Inspect	Inspect	SA-ADM...	2.20.36.50		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	tiqcdn.com HTTPS Inspected
Today, 1:12:08 PM	Firewall	Mgmt	SA-FWL-001	Accept		SA-ADM...	a23-51-123-27.deploy.static.akamaitechnologies.com (23.51.123.27)		http (TCP/80)	Admin CA (Admi...	sa-adm-01@salte...	http Traffic Accepted from Admin CA (AdminCA)(10
Today, 1:12:07 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Bypass	Bypass	SA-ADM...	62.138.155.102		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	www.askus.hsbc.co.uk HTTPS Bypassed
Today, 1:12:07 PM	Firewall	Mgmt	SA-FWL-001	Accept		SA-ADM...	91.214.6.22		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	https Traffic Accepted from Admin CA (AdminCA)(1
Today, 1:12:07 PM	Firewall	Mgmt	SA-FWL-001	Accept		SA-ADM...	91.214.6.22		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	https Traffic Accepted from Admin CA (AdminCA)(1
Today, 1:12:06 PM	Firewall	Mgmt	SA-FWL-001	Accept		SA-ADM...	91.214.6.22		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	https Traffic Accepted from Admin CA (AdminCA)(1
Today, 1:12:06 PM	Firewall	Mgmt	SA-FWL-001	Accept		SA-ADM...	91.214.6.22		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	https Traffic Accepted from Admin CA (AdminCA)(1
Today, 1:12:06 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Bypass	Bypass	SA-ADM...	91.214.6.22		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	www.hsbc.co.uk HTTPS Bypassed
Today, 1:12:06 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Bypass	Bypass	SA-ADM...	91.214.6.22		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	www.hsbc.co.uk HTTPS Bypassed
Today, 1:12:06 PM	Firewall	eth2	SA-FWL-001	Accept	Bypass	SA-ADM...	91.214.6.22		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	https Traffic Accepted from Admin CA (AdminCA)(1
Today, 1:12:06 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Bypass	Bypass	SA-ADM...	91.214.6.22		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	www.hsbc.co.uk HTTPS Bypassed
Today, 1:12:06 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Bypass	Bypass	SA-ADM...	91.214.6.22		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	www.hsbc.co.uk HTTPS Bypassed
Today, 1:12:06 PM	Firewall	Mgmt	SA-FWL-001	Accept		SA-ADM...	ec2-54-83-28-177.compute-1.amazonaws.com (54.83.28.177)		http (TCP/80)	Admin CA (Admi...	sa-adm-01@salte...	http Traffic Accepted from Admin CA (AdminCA)(10
Today, 1:12:06 PM	Firewall	Mgmt	SA-FWL-001	Accept		SA-ADM...	ec2-54-83-28-177.compute-1.amazonaws.com (54.83.28.177)		http (TCP/80)	Admin CA (Admi...	sa-adm-01@salte...	http Traffic Accepted from Admin CA (AdminCA)(10
Today, 1:12:07 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Bypass	Bypass	SA-ADM...	91.214.6.22		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	www.hsbc.co.uk HTTPS Bypassed
Today, 1:12:07 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Bypass	Bypass	SA-ADM...	91.214.6.22		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	www.hsbc.co.uk HTTPS Bypassed
Today, 1:12:07 PM	Firewall	eth2	SA-FWL-001	Accept	Inspect	SA-ADM...	2.20.36.50		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	https Traffic Accepted from Admin CA (AdminCA)(1
Today, 1:12:07 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Inspect	Inspect	SA-ADM...	2.20.36.50		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	tiqcdn.com HTTPS Inspected
Today, 1:12:04 PM	Firewall	eth2	SA-FWL-001	Accept	Inspect	SA-ADM...	ams15s33-in-f3.1e100.net (172.217.20.67)		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	https Traffic Accepted from Admin CA (AdminCA)(1
Today, 1:12:04 PM	HTTPS Inspection	Mgmt	SA-FWL-001	HTTPS Inspect	Inspect	SA-ADM...	ams15s33-in-f3.1e100.net (172.217.20.67)		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	www.google.co.uk HTTPS Inspected
Today, 1:12:04 PM	Firewall	eth2	SA-FWL-001	Accept	Inspect	SA-ADM...	ams15s33-in-f3.1e100.net (172.217.20.67)		https (TCP/443)	Admin CA (Admi...	sa-adm-01@salte...	https Traffic Accepted from Admin CA (AdminCA)(1

So my conclusions are:

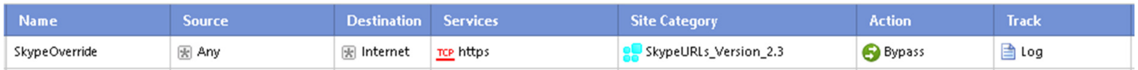
1. **Don't use probe bypass** and your HTTPS inspection will be MUCH more reliable for much more of the Internet. In this state, URL resource based (and category based) HTTPS exceptions (specifically those for sites and apps with any URL on an SNI based server or service) will be possible to make work as well!
2. **One can meet the requisite privacy obligations perfectly well without probe bypass enabled.**

Now back to Skype:

A custom application with a long list of URLs

For Skype we must create a custom application and import these URLs and add this to the HTTPS inspection policy as an override 'bypass' above the HTTPS inspection rule(s). Don't forget to save the HTTPS inspection policy and close it before installing the policy on to the gateway(s) – it's a bit unreliable (crashes etc.) handled any other way.

NB: This must be just a 'custom application' Don't recycle the 'SkypeALL' groups used in the Access Policy – at the time of writing (R80.10) this is not the correct approach.



Name	Source	Destination	Services	Site Category	Action	Track
SkypeOverride	Any	Internet	TCP https	SkypeURLs_Version_2.3	Bypass	Log

Providing we have Probe Bypass disabled, and we are on the latest (GA only of course) Jumbo hotfixes (Jumbo 103 for R80.10, Jumbo 302 for R77.30 at time of writing) I have compiled a list of URLs for both Skype for Business and Skype consumer edition. Tested at version 7.41.0.101 consumer and Skype for business (Lync) 2015 Version 15.0.5031.1000.

These URLs are all in a file named 'SkypeURLs Version 2.3 August 2018'.XLSX.

After this list of URLs there are the ESSENTIAL requirements at a port level discussed below.

Please note that there are various definitions of which high ports Skype requires, including the ports in the new Check Point Skype application object. That application object includes more than I have found to be required; the ports and services below have proved to be successful in all cases so far – in some we have 3 -4 months of successful testing.

My URL list is at version 2.3 as at 31st August 2018 and is the one we are using. There may be the odd redundancy in here, but don't be fooled into thinking that '*.pipe.aria.microsoft.com' would cover 'pipe.aria.microsoft.com' nor that *.live.com would cover odc.officeapps.live.com. It is true that *.officeapps.live.com would cover odc.officeapps.live.com. but I prefer to be explicit. This is why our list has a load of x.skype.com stuff in it so we can see what we are doing and why. I have included each of the x.skype.com items as well as *.skype.com. Previously I did not have*.skype.com so I could monitor for new URLs but it was proving to be a pain. I might take the time to turn all of this in to some REGEX with a bit of future proofing in it to address all of the above.

Important Outbound services for Skype

There are some *OUTBOUND* ports that we that are (more or less) essential for Skype to work properly.

For Skype to work well we need to allow these in the access policy. Skype will probably work sometimes without them but to make it connect quickly and be reliable for test chat and calls these are very much a requirement. **All four of these services or ranges only need to be allowed outbound from the LAN, WIFI etc.**

TCP 5061

SIP_TLS_Authentication – It's pretty obvious what this is, and Skype uses it

UDP 3478-3481

This is the STUN (Session Traversal Utilities for NAT) suite ports which allow for negotiation of VoIP tunnels behind NAT devices. Skype *may* find a way to work without this but you don't want it to have to try.

UDP 50000-59999 (September update – increased the top of this range to 65535)*

&

TCP 50000-59999 (September update – increased the top of this range to 65535)*

I observed connections from the Skype test clients on ports higher than the Skype published 59999 port.

I'm usually reluctant to open up big ranges like these but I'm a little more relaxed in this case because the Skype process for text chat, calls (and video calls) and screen sharing is (more or less):

After an initial contact and exchange of details (IP addresses etc)

1. **Try to connect directly to the LAN IP address of the recipient/partner. If this is an RFC 1918 address this will, of course only work on the LAN/WAN but not over the internet. This is very good for fast connections between computers on corporate networks. This will be on the high ports above 50000-65535. It has always been TCP when I have observed it but Microsoft say that it can be UDP too.**
2. **Try the same process on the high ports using the NAT IP address of the device**
3. **For VoIP calls (and video) make a linkup between the NAT IP addresses of the two devices using the STUN protocol**
4. **If all of the above fails, attempt to get something together using just TLS connections to the various Skype server systems from both partners in the call including (but not limited to) incapsula.com, secure.aadcdn.microsoftonline-p.com, sipfed.online.lync.com (possibly this last one is Skype for business only) etc. As you can imagine, this is less likely a) to work and b) to be a reliable and quality connection**

So – I am allowing these ports both TCP and UDP outbound from workstations or networks where we wish to use Skype and want it to work well. It's not a big security issue, these are outbound only and if we are allowing Skype and all of its abilities on a given network it's clear we're not much concerned with data exfiltration! The risk of some malware using these ports in future cannot be ignored but as we are not using the probe bypass any more, the initial connection to a C&C IP address should be recognised by the anti-bot blade anyway. The only concern is that we are, buy this whole process, not going to be able to do any threat emulation or malware detection on files (or HTML links) passing though Skype – cue discussions of sandblast agent…another time…

Finally, one remaining issue:

Having done all of this, I can see no further URLs being inspected as part of the connectivity and no broken SSL/TLS connections; however in Skype for business I cannot successfully paste a PNG file (snipping tool screenshot) into the chat window. It tries and then fails. September addition – I have just tried this between a Skype non-commercial and a Skype for business which are on the same LAN and have Internet access with no HTTPS inspection and it *still* fails! I think this might be similar to the inability to use screen-sharing in that scenario and would probably work on business to business or non-commercial to non-commercial.

One last note:

On one site where we went live with all of this today, it worked for some users and not for others. We discovered two interesting little 'gotchas' which made the above 'seem' unreliable until we resolved them. I am documenting them here to assist with demonstrating that when something doesn't quite work – it is not always just the thing we are testing at that time, other factors must be considered…

1. Our override group was not being applied to the same source group of users and computers as the actual 'inspect' rule at the end of the rulebase, this meant that some users did not have the overrides at all! So we learned here to always ensure that overrides are *above* the inspect rule in the rule base (we knew that already of course) *and* **that the override rule(s) apply to the same groups of users** - or at least those that you want to have said overrides!
2. This was a hard one to notice: Some users still found Skype was not working properly. We noticed in the logs a lot of different connections attempting to be made to different URLs than for those computers where it had worked fine. Notable among these was ***sipfed.online.lync.com*** (this example comes from an end user who uses Skype for business). We eventually (the end user did to be fair) figured out that the thing they all had in common was that they were laptops and that laptops had a desktop firewall policy which was not allowing the UDP 3478-3481 STUN connections or the TCP & UDP 50000-59999 outbound connections. Of course, we would not see these attempting and failing in the Check Point logs because we are only looking at the logs for the gateway firewalls. So all we see is the Skype client behaving differently. It attempts the above connections, fails and so goes for the less reliable 'sipfed' based solution. – which incidentally is another SNI based URL!