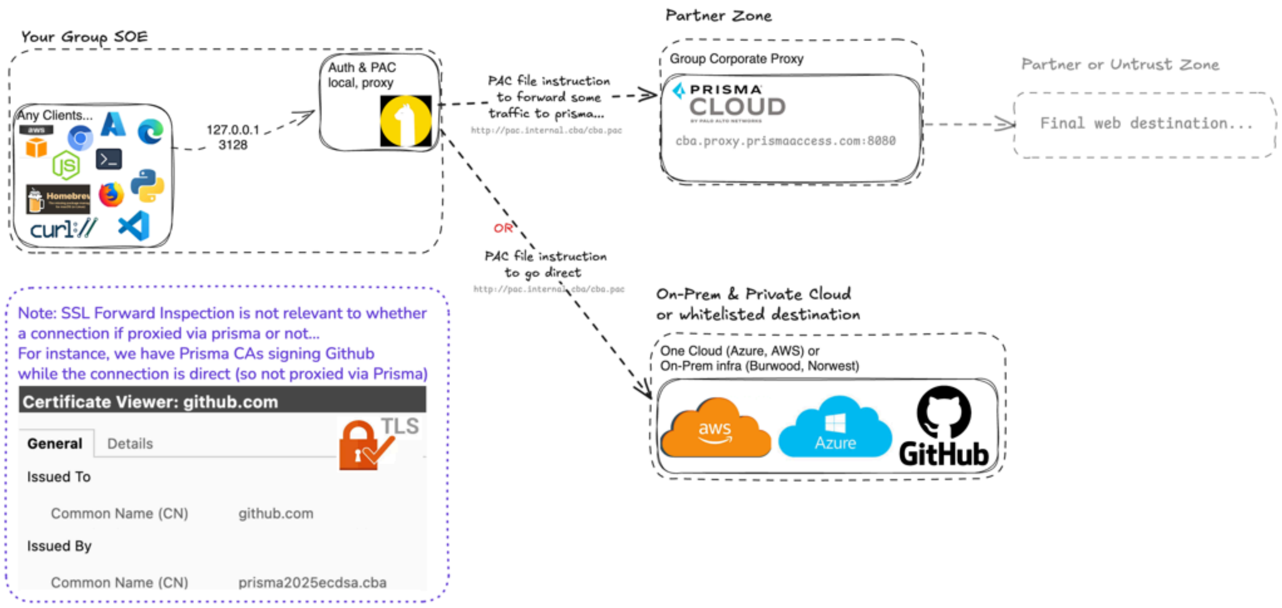
🐪 Setting up Alpaca on Windows

I recently updated this guide for macOS. Due to high demand from many people reaching out via MS Teams,
I realised a guide for Windows is also much needed!

Table of Contents:

Introduction



Note: SSL Forward Inspection is not relevant to whether a connection if proxied via prisma or not...
For instance, we have Prisma CAs signing Github while the connection is direct (so not proxied via Prisma)

Certificate Viewer: github.com

General | Details

Issued To

Common Name (CN)	github.com

Issued By

Common Name (CN)	prisma2025ecdsa.cba

If you on MacOS, head this way → 🐫Setting up Alpaca on MacOS
Always try Self-Service first, as it has a dedicated team for better support.

Why writing a MacOS guide if Self-Service offers it?
At the time of writing the MacOS article, for the MacOS SOE (not the EPZ one), JAMF On-Prem (the former Self-Service) **did** not have a functional Alpaca: It did not detect the PAC URL (I fixed that in 2.0.11) so all connections defaulted to DIRECT (where 95% should be forwarded to Prisma proxy). Then JAMF Cloud (the new/current Self-Service) **did** not have Alpaca packaged… but it now has!

This guide is based on my findings while assisting Group staff with Windows, as I do not have access to a Windows SOE for testing: The main issue I want to address is how to auto-start Alpaca on Windows, but I haven't tested it yet. If anyone is interested, I would appreciate any assistance. We just need a Group Windows SOE.

Why am I promoting Alpaca on our SOE?
The answer lies here: CLI - Why you should not set CNTLM nor Prisma

Alpaca uses a PAC URL/File that automatically detects where to route web traffic:

- Directly, for sites like github.com , most of *.cba, and some Microsoft domain names…
- Or via a proxy, currently `http://cba.proxy.prismaaccess.com:8080`

Convinced ?
If not, that's all right, then you can set `http://cba.proxy.prismaaccess.com:8080` for `http_proxy` and `https_proxy` environment variables and maintain the value of `no_proxy` .
😭"no_proxy" variable (on Group SOE)
Otherwise the guide below will guide you throught using Alpaca so you don't need to mock around the `no_proxy` variable.

Requirements

- Application Whitelisting so you can run the executable from C:\Tools or C:\Dev without an agent or GPO blocking it…
  See this https://engineering-handbook.pages.commbank.io/knowledge-base/windows-soe-guides/app-control
- Admin access is preferable but is ‘not’ a must…

Downloading Alpaca…

The first challenge is that the downloaded executables are corrupted (0KB). I suspect our Palo Alto firewalls may be the cause (Maybe Wildfire blocking unknown / new PE?).
Downloading `alpaca*.exe` from Releases · samuong/alpaca will likely fail.

Instead, you can download it from this link: alpaca_v2.0.11_windows-amd64.exe. This file is hosted on my OneDrive, which is far from ideal, but it's the only working solution I have.
Integrity checks:
`sha256:3436282970b3df2d8d311f36c829b9e9bd58669930ab36105b86a5d33dc9498c`

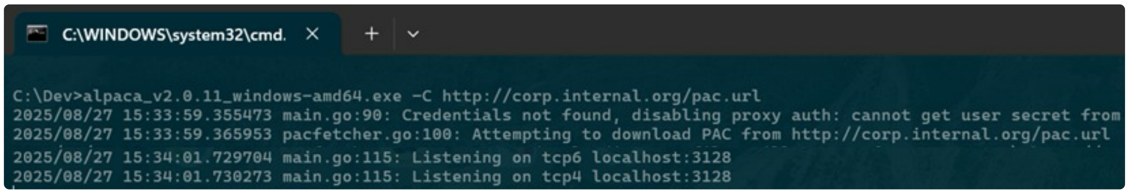
Running Alpaca…

Place the exe in App Whitelisted Excludion folders in C:\Dev **after** requesting that access

I'm afraid, like (it was) on macOS, Alpaca does not auto-detect or pull the PAC URL from the SOE 😞! Disappointing…
We will need to manually specify it. Create a batch file (also in C:\Dev) similar to the one below:

```
@echo off
C:\Dev\alpaca_v2.0.11_windows-amd64.exe -C http://pac.internal.cba/cba.pac
```

Run the batch (e.g. create a shortcut to the Desktop) and you should see something like:



```
C:\Dev>alpaca_v2.0.11_windows-amd64.exe -C http://corp.internal.org/pac.url
2025/08/27 15:33:59.355473 main.go:90: Credentials not found, disabling proxy auth: cannot get user secret from
2025/08/27 15:33:59.365953 pacfetcher.go:100: Attempting to download PAC from http://corp.internal.org/pac.url
2025/08/27 15:34:01.729704 main.go:115: Listening on tcp6 localhost:3128
2025/08/27 15:34:01.730273 main.go:115: Listening on tcp4 localhost:3128
```

A few more things…

NTLM authentication

If you want/need NTLM Credentials, read this → [GitHub - samuong/alpaca: A local HTTP proxy for command-line tools. Supports PAC scripts and NTLM authentication.]

Cannot bind to 3128 ?

You prob have CNTLM installed with a Service (auto-start). So you'll need to kill the process, stop the service or uninstall CNTLM

Using Alpaca…

Now that Alpaca is running and ready to forward web requests to the right locations, we still need to tell your various applications to use it…

When not to use Alpaca

By default, it something works, just don't change its settings… Some applications like your web-browsers would already make use of the PAC file so they do not need Alpaca… other just have the right settings already!

When to use Alpaca

If an application doesn't work however and output some connection errors (Notably HTTP 503, or `connection error`), you're likely needing to add, remove or edit your settings, possibly including environment variables:

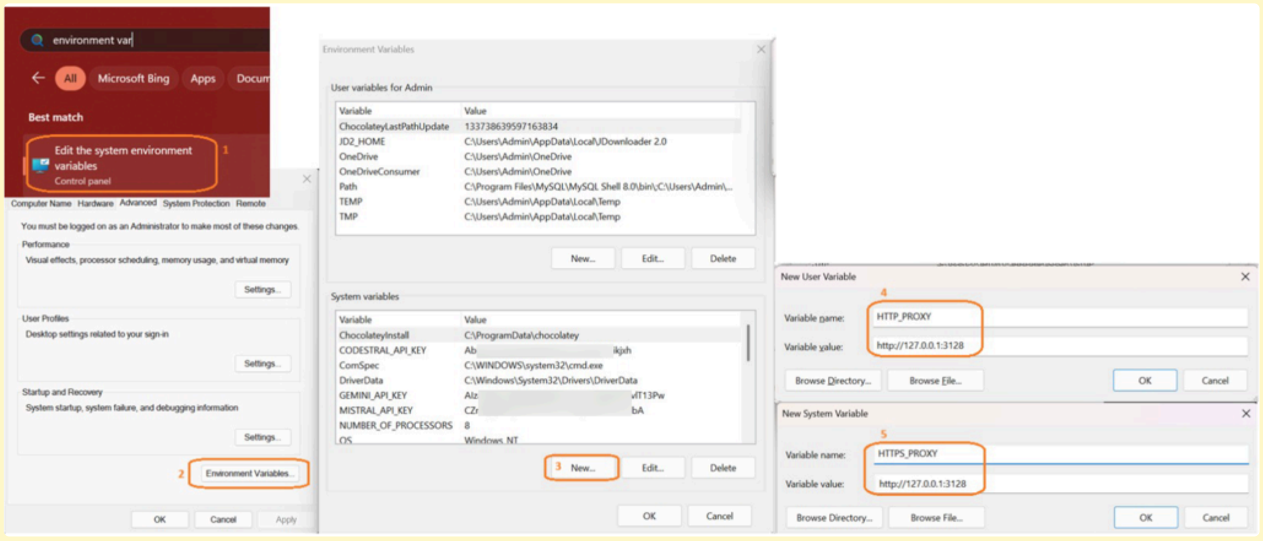
Application Specific

Environment Variablest nve

Most CLI (for command-line interface) require `http_proxy` and `https_proxy` to be set to `http://127.0.0.1:3128` **only** if you have Alpaca up and running (*see cmd screenshot above….*)

⚠ Changing **System** environment variables requires admin access. Without it, you cannot create those variables.
You might be able to create **User** environment variables, but **System** variables may take

precedence if they overlap.
Conflicting System and User variables can disrupt your system!



You must be logged on as an Administrator to make most of these changes.

Performance
Visual effects, processor scheduling, memory usage, and virtual memory

User Profiles
Desktop settings related to your sign-in

Startup and Recovery
System startup, system failure, and debugging information

User variables for Admin

Variable	Value
ChocolateyLastPathUpdate	133738639597163834
JD2_HOME	C:\Users\Admin\AppData\Local\JDownloader 2.0
OneDrive	C:\Users\Admin\OneDrive
OneDriveConsumer	C:\Users\Admin\OneDrive
Path	C:\Program Files\MySQL\MySQL Shell 8.0\bin\;C:\Users\Admin\...
TEMP	C:\Users\Admin\AppData\Local\Temp
TMP	C:\Users\Admin\AppData\Local\Temp

System variables

Variable	Value
ChocolateyInstall	C:\ProgramData\chocolatey
CODESTRAL_API_KEY	Ab[redacted]ikjxh
ComSpec	C:\WINDOWS\system32\cmd.exe
DriverData	C:\Windows\System32\Drivers\DriverData
GEMINI_API_KEY	AIz[redacted]vIT13Pw
MISTRAL_API_KEY	CZr[redacted]bA
NUMBER_OF_PROCESSORS	8
OS	Windows_NT

New User Variable
Variable name: HTTP_PROXY
Variable value: http://127.0.0.1:3128

New System Variable
Variable name: HTTPS_PROXY
Variable value: http://127.0.0.1:3128

To complicate matters...

You can set `http_proxy=http://cba.proxy.prismaaccess.com:8080` and `HTTP_PROXY=127.0.0.1`. Most clients read these variables in both upper and lower case, so users must ensure they are not declared in conflicting ways. The application's response to such conflicts is uncertain.

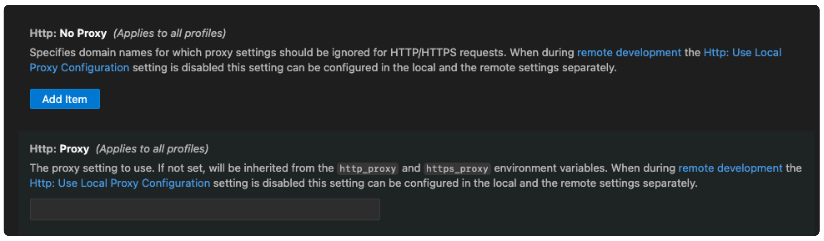
Python, curl, Node, npm, pip, aws-cli, az, VSCode and others will check for the existence of environment variables and use them if available.

App Settings

If an application fails to connect, it is either:

- using a dead proxy. e.g. if Alpaca is not running, using http://127.0.0.1:3128 will inevitably fail!
- setting the wrong environment variables. e.g typos in the value
- or it has *wrong* custom settings taking precedence over these environment variables

For instance, for VSCode, you have the opportunity to set proxy and no_proxy variables in `Preferences / Settings / Proxy`


Http: No Proxy *(Applies to all profiles)*
Specifies domain names for which proxy settings should be ignored for HTTP/HTTPS requests. When during remote development the Http: Use Local Proxy Configuration setting is disabled this setting can be configured in the local and the remote settings separately.

Add Item

Http: Proxy *(Applies to all profiles)*
The proxy setting to use. If not set, will be inherited from the `http_proxy` and `https_proxy` environment variables. When during remote development the Http: Use Local Proxy Configuration setting is disabled this setting can be configured in the local and the remote settings separately.

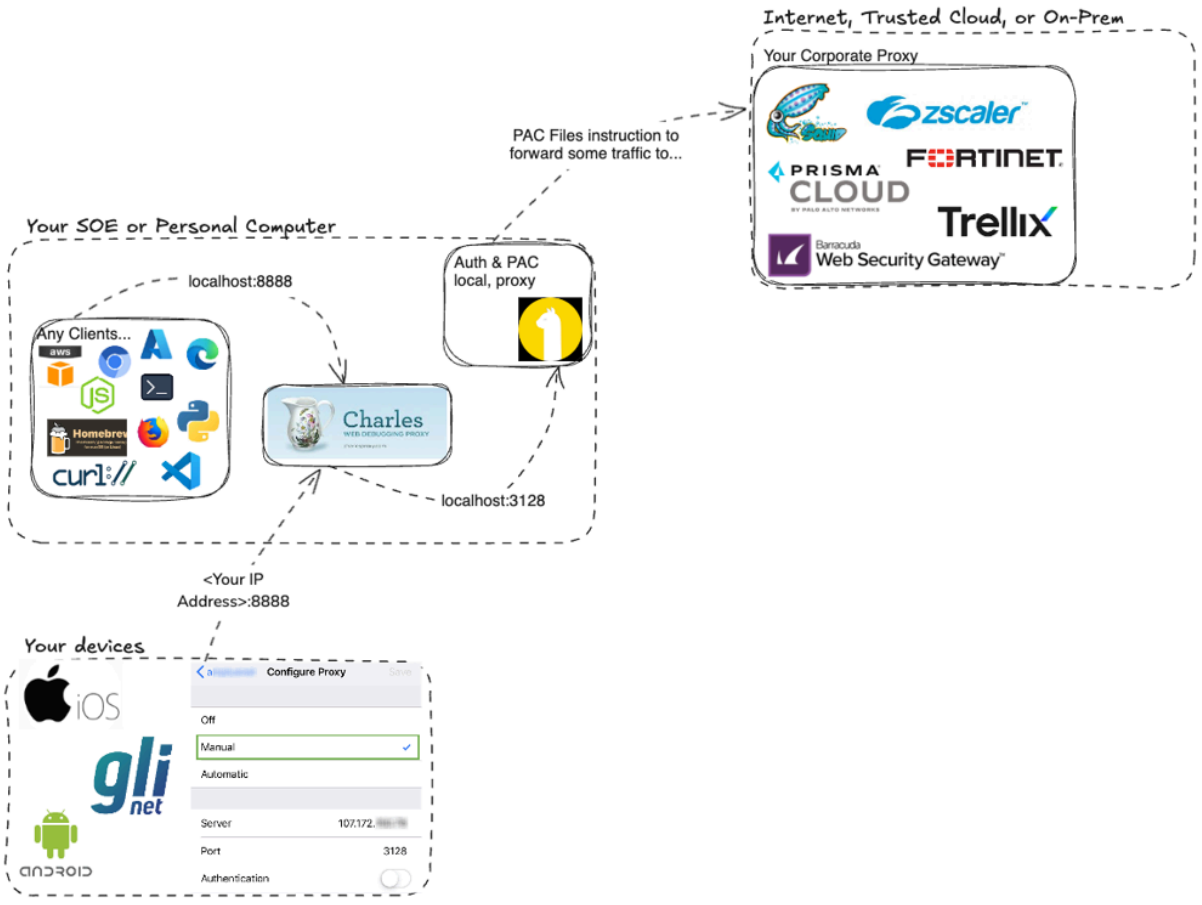
Note: mine are not set, because VSCode is using the Env Variables I have set!

to make things more complicated...

Occasionaly however, I set these settings to another proxy, named Charles, for security and testing purposes (Web Debugging)

In fact, Charles can be used to solve a pesky problem with `Python 3.13` and our non RFC-compliant signing CAs...

So App Settings and Env Variables are not mutually exclusive! You could set the `Env Var` for all Apps and Clients to use Alpaca, while having custom settings for a particular App like VSCode going to another proxy like Charles or Fiddler for inspection...



Improvments

Autostart

On macOS, we have this capability, but not on Windows.

I welcome any assistance (I don't have a Windows SOE) or insights on how to run Alpaca silently at user logon or pre-logon (system).

Tweaking `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run` to point to the batch file, or using Scheduled Tasks, should accomplish this easily.

The batch file can be adjusted to run in the background and terminate the process listening on port 3128 to restart it with each execution.

Group-wide deployment / Strategic Plan

I ~~am~~ was a team lead for the Cyber Retail Tech and Bankwest team and for a GenAI practice tasked with building an AI augmented Threat and Risk assessment tool. Since joining CommBank in 2020, I have noticed proxy and SSL issues affecting productivity. These issues hinder productivity for both engineers and non-engineers, prompting me to address them to improve efficiency and reduce troubleshooting time

The provisioning team for MacOS is packaging Alpaca via JAMF/Self-Service.

The SCCM team (Software Center) should adopt a similar approach for Windows machines to:

- Run Alpaca as a service
- Include the certificate store (cacert.pem) with pre-built internal and external CAs
- Set environment variables to point to it

If you are part of this team (EUX/SCCM) and want to implement and test the above, I am happy to assist, as this would enhance productivity significantly.

Conclusion

I hope you have a working proxy setup on your SOE, whether on a MacOS or Windows machine, and whether it's the standard SOE or an EPZ. While I cannot address all proxy and SSL issues for the entire enterprise, I welcome your questions or requests for assistance. I will continue to create and update these guides to reduce the need for manual or interactive support sessions.

Note, if you have issues pertaining to SSL / TLS trust, this wouln’t be proxy related, and the fix can be found here: https://engineering-handbook.pages.commbank.io/knowledge-base/windows-soe-guides/CBA-root-certificate

Helpful? Drop me a thanks on Achievers !
And if you've got knowledge to share, don't hold back - we all grow when we learn from each other 💡