I was tasked to do a penetration test of a /16 network and was told to focus on the web applications. After a discovery scan returned ~15,000 hosts, I really just wanted to focus on systems that were potentially running some sort of web services. A neat little trick in nmap is to just scan by a particular service, so I ran the command nmap -p http\* -iL discover\_hosts.txt, which returned about 4,000 machines that were running at least one service on those ports. Around 3,000 were listening on either port 80 or 443. I randomly browsed to a few of these sites and found a lot of boring printers and forgotten Apache test pages, but I really needed something to help organize and help prioritize these web targets, so I coded a few tools to make this easy.

The first tool I wrote is a Python script, sitekick.py. It is multi-threaded, web scanner that takes a list of URLs and sends web requests to each target. If the target responds, Sitekick parses the title of the site, records the server header, and outputs everything to a csv file or the terminal (or both). For sites that do not render unless JavaScript is detected, Sitekick will use selenium and PhantomJS to send a silent browser to the target to gather the info. It’s a cool tool, but that’s not what this post is about…this post is about how I wrote a similar tool in PowerShell!

The cmdlet I wrote, Get-WebsiteInfo, was my first experience with PowerShell runspace pools (multithreading). The most helpful thing that I read was this: <https://www.codeproject.com/Tips/895840/Multi-Threaded-PowerShell-Cookbook>. It included relevant examples and really helped me understand how to implement multi-threading in PowerShell.