An Introduction to “Scan Everything”

At Bit Discovery, we often must walk both clients and potential clients through the rational objection to the idea of adding everything to their inventory and then testing everything that they find. The concern is understandable – it’s expensive, creates duplicate workload, potentials for false positives grow, and any additional risk of introducing accidental load that might cause a denial of service. I don’t want to discount the real and rational objections; however, I will spend some time dispelling these objections as a good security practice.

As anyone who knows Jeremiah and my background knows, we spent a lot of time in the webappsec trenches. I have personally performed countless penetration tests, and WhiteHat was one of, if not the largest, SaaS-based testing platforms in existence. That’s has led us to find a great deal of the oddities of the Internet through trial and error. Let me guide you through a few such issues.

1. Flappers.

When you have two or more machines that are load-balanced (typically through round-robin DNS for the purposes of this example), there is always the possibility that one or more of them will have issues. Just think about it practically. Let’s say you must apply a patch manually for some reason. Are you going to apply them all at the exact same millisecond? Unlikely. You’re more than likely going to have a slight disparity between the machines, even if briefly. Even if you do it in an automated fashion, a tiny load on one machine might delay a patch by a few seconds. Now amplify that by forgetting to install a patch on one machine or one machine being misconfigured. Or worse, one machine could be hacked. If you only look at one machine, you are leaving a massive gap in the reality of how your architecture is set up. Because you don’t make the distinction between these machines and instead group them together, you are very likely going to miss flappers entirely, or if you do find them, it will be by luck or take a long time for the naive scanner to find the IP through round-robin DNS to test it.

2. Overly De-Duped Infrastructure.

We often see that customers are wary of scanning the same machine over and over for denial of service, so they spend a lot of time trying to de-duplicate their environment. Let’s take an example of two websites (dev.whatever.com and prod.whatever.com) where the HTML is a perfect match. Are they the same websites? Well, one could argue that the code is identical or at least will be identical whenever prod catches up with dev. But dev is almost always going to be out of sync with prod. Meanwhile, dev is not behind the cloud-based web application firewall for cost reasons, so it has minimal protection, next to no logging, and hasn’t received a patch since dinosaurs roamed the earth. But is dev any less dangerous? If it’s behind the same firewall and has the same username and pass to the database (just a different named database (dev.db.int vs. prod.db.int), that won’t do much to stop an attacker who wants to pivot within a network. And who protects internal machines anyway? If you aren’t scanning your external presence thoroughly, you almost certainly aren’t doing a good job internally. When you do this kind of de-duplication, you lose a lot of signal, which can obscure what’s really going on.

3. Ignoring Assets that “Don’t Matter.”

This is a very subjective issue – what matters? Well, the simple answer is, no one

knows what matters for sure until after it's too late. You may have a pretty good inkling of the things referred to as "alpha assets" – the assets that provide the bulk of your income. But the long tail of asset value is very long and too complex for anyone to do manually. If you spend all your time focused on assets that you do know are important but don't correctly deal with the long tail, either by losing them over time or willful ignorance, you will likely find the true value of those assets. Equifax found out their dev site existed too late to patch Apache Struts. Sands Casino found out their dev site was a conduit to production. Verizon found out their router was publicly accessible after the fact. The examples continue to grow, so there should be no more excuses for any qualified person in infosec not knowing the value of the long tail.

4. Using Appsec Scanners in lieu of Asset Management.

I have seen a number of companies that think that anything they scan is the thing that matters (self-evidently true from their perspective) and that they rely on the scanners to find any and all issues. The problem is different scanners are good at different things. I wouldn't use a port scanner to find web app issues and vice versa. Likewise, if you are only scanning some small percent of your network, you are likely missing out on a huge number of assets when a zero-day comes out. Sure, the population of that zero-day exploit may only exist in a small number of machines you are testing, but how many machines aren't you testing that also suffer the same issue? How could you know unless you are looking at everything all the time? I would never claim that asset management is a substitute for a vulnerability scanner or penetration test, but likewise, I wouldn't say that either of those is good at finding issues in assets that aren't under contract.

"You cannot accurately quantify your management of corporate risk when you have no idea what that risk is."

So yes, it's expensive to find, test, triage all your assets, but to do so means you finally have some idea of what the risks are and can prioritize your risks and respond appropriately. **The answer, therefore, is in finding less expensive ways to audit more, NOT to audit less.** If you don't test your environment thoroughly, you really will have no idea how good or bad your environment is. Said another way, **you cannot accurately quantify your management of corporate risk when you have no idea what that risk is**.

POST BY ROBERT HANSEN

SEPTEMBER 14, 2021

CONNECT WITH US

RELATED POSTS