F5 Friday: You&rsquo;ll Catch More Bees with Honey(pots)

Lori MacVittie, 2010-22-10

Catching bees with honey(pots) means they're preoccupied with something other than stinging you.

Pop quiz time…pencils ready? Go.

Is it good or bad to block malicious requests?

If your answer was "that depends on a lot of different factors" then pat yourself on the back. You done good.

It may seem counterintuitive to answer "it's bad block malicious requests" but depending on the attacker and his goals it may very well be just that.

MISSION IMPOSSIBLE

No security solution is a 100% guaranteed to prevent a breach (unless we're talking about scissors) and most are simply designed to accomplish two things: buy you time and collect enough information that you can address the underlying vulnerability the attacker is attempting to exploit. Some solutions buy you more time than others, and some solutions provide the ability to collect more data than others, but in the end an attacker – like an application developer - with enough time and money and information **will** find a way to breach security. This is particularly true for new vulnerabilities and attack methodologies with which infosec professionals may be not familiar because, well, they're newly discovered (or pre-discovered – someone has to be victim number one, after all) and there just isn't a lot of information about it yet.

Now, the reason that blocking those malicious requests could actually be serving the miscreant is that over time, a motivated attacker can learn a lot from the security solution, including how it works and what it's specifically protecting. It can take weeks, but over time the attacker can build a profile of your security infrastructure based on the blocking of requests (mapping parameters and values and paths that caused the request to be blocked) and subsequently find a way around it. This is true regardless of whether the blocking mechanism is implemented in the application itself or in network-deployed security infrastructure.

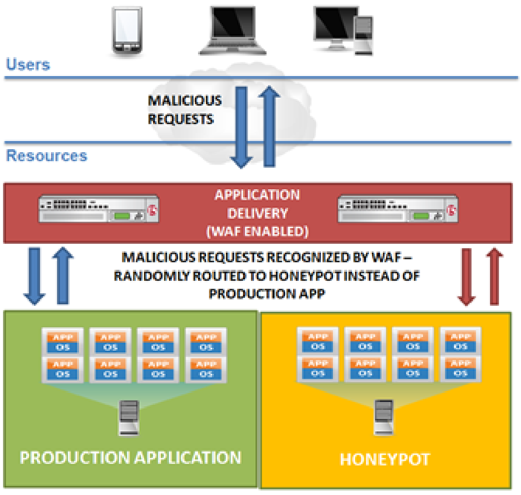
Your new mission then, should you choose to accept it, is to confuse the attacker for as long as possible, essentially buying you time to figure out what they're trying to do. Then you can patch or deploy or notify the proper authorities and try to put a stop to the attacker as well as the attacks.

One of the ways in which you can buy a lot more time for researching and implementing a solution against old or new attack methodologies is to employ a strategy that combines a WAF (web application firewall) and a honeypot.

NOT POOH BEAR'S HONEYPOT

In almost every story about Pooh Bear he complains about a "rumbly in his tummy" and then laments the fact that his honeypot is nearly empty. The honeypot you want to leverage is one that Pooh Bear would love: it automatically reloads itself to an untouched state on a specified interval. Virtualization has afforded organizations the ability to easily implement honeypots that are exact duplicates of production applications and keep them "pristine" across time by reloading the original virtual image.

That comes in handy when it comes to confusing attackers. Imagine their frustration when their last attack appeared to be successful, depositing a file on the web server, and when they try to access it, it isn't there. Ha! Good times, good times.


Users
MALICIOUS REQUESTS
Resources
APPLICATION DELIVERY (WAF ENABLED)
MALICIOUS REQUESTS RECOGNIZED BY WAF – RANDOMLY ROUTED TO HONEYPOT INSTEAD OF PRODUCTION APP
APP OS
PRODUCTION APPLICATION
HONEYPOT

But in order to accomplish this frustrating and protective strategy you first must have deployed a WAF capable of detecting an attack in progress (hint: it also must be deployed in reverse-proxy mode). And it has to be fairly accurate because you really don't want to route legitimate users to a honeypot. That'd be frustrating, too, but you'll get calls about that one. I can almost guarantee (with Heisenberg certainty) that an attacker won't call you even if they do figure out they're being routed to a honeypot.

F5 BIG-IP Application Security Manager (ASM) can do this thing. Using a combination of techniques it can, with good accuracy, determine when the applications it protects are being attacked. It does so through a combination of inspecting the client, the requests, and the patterns of those requests. Once it is determined that it is under attack it raises an event in the underlying, shared application delivery platform (TMOS) that can be acted upon using F5's network-side scripting technology, iRules.

Using iRules you can do, well, just about anything – including randomly routing requests to a honeypot. The reason I say "random" is that any consistent reaction to a motivated attacker gives them more information upon which they can act to circumvent the security systems. Like timing-based attacks, one of the ways to successfully avoid compromise is to randomly change the response pattern. A simple approach would be to decide that one of every X requests will be randomly routed to the honeypot. Additionally you'd want to apply a rate-limiting policy to the attacker to ensure their attacks don't overwhelm legitimate traffic.

This approach impedes the ability of the attacker to consistently gather information about the underlying architecture and security infrastructure that can be used against you. Such a strategy may in fact hold off the attacker indefinitely, although there are no guarantees. More likely it's just buying you even more time in which you can gather forensic evidence for the authorities (because you are doing that, right?) and figure out if there is, in fact, a vulnerability for which a solution exists and can be applied before it is exploited in your environment.

This approach also works to mitigate bots (web scrapers). Yes, you could – upon detection – simply close their sessions but they'll just open new ones. Yes, Javascript-based protections can usually detect a bot versus a human being, but it – like all security solutions – is not 100% foolproof. So instead of letting the web scraper know they've been caught, direct them to an application in the honeypot that contains a lot of irrelevant data. Assign them to a low rate class to limit their potential impact on the performance of the application, and let them download like there's no tomorrow. Imagine their faces when they realize they've spent hours scraping what turns out to be useless data! Ha! Good times, good times.

AGILE SECURITY is a PART of an AGILE INFRASTRUCTURE

The ability to determine how best to respond to an attacker using network-side scripting is unique to BIG-IP ASM. The underlying integration with the underlying unified application delivery platform makes it possible for security professionals to take advantage of the core traffic management capabilities available on the BIG-IP platform, such as network-side scripting and rate shaping. Yes, you can leverage standard policies if you like, but the ability to customize if/when necessary makes your entire security infrastructure more agile; it affords the opportunity to respond to attacks and vulnerabilities on-demand without requiring modification to applications or the rest of the infrastructure.

Combining the flexibility of virtualization, which provides an affordable mechanism for deploying a mirror image (pun intended) of production apps and thus building out a honeypot, with the ability to dynamically and flexibly route requests based on context atop the capability to detect the complex attack patterns applications are increasingly subjected to makes it possible to better protect data center resources without compromising availability or performance for legitimate users.

F5 Networks, Inc. | 401 Elliot Avenue West, Seattle, WA 98119 | 888-882-4447 | f5.com

F5 Networks, Inc.
Corporate Headquarters
info@f5.com

F5 Networks
Asia-Pacific
apacinfo@f5.com

F5 Networks Ltd.
Europe/Middle-East/Africa
emeainfo@f5.com

F5 Networks
Japan K.K.
f5j-info@f5.com