Bot Defense

Insights Into Basic And Advanced Techniques For Thwarting Automated Threats

An ENTERPRISE MANAGEMENT ASSOCIATES® (EMA™) White Paper

October 2016

EMA™

IT & DATA MANAGEMENT RESEARCH,
INDUSTRY ANALYSIS & CONSULTING

Bot Defense: Insights Into Basic And Advanced Techniques For Thwarting Automated Threats

Table of Contents

Bot Defense: Insights Into Basic And Advanced Techniques For Thwarting Automated Threats

Executive Summary

This paper lays the foundation for what bots are, why they are a threat, how they are adapting to bypass traditional defense technology, and how to overcome those shortcomings. Most current solutions were designed to combat programming deficiencies and errors, such as those addressed in the Open Web Application Security Project (OWASP) Top 10 and SANS/Mitre Top 20, but are not sufficient to protect them from exploits in business logic.

Organizations hoping to gain the best defense from these newly-defined OWASP Automated Threats to Web Applications must understand and then adopt an adaptive defense that uses both older and new defense capabilities.

What Is A Bot?

To start the discussion, it is appropriate to set the foundation of what a bot is and why they are an issue to Internet-connected environments. The term "bot" is short for robot, but when used in the context of the Internet, the full term is Internet robot or Web robot. In their basic form, bots are computer systems that are programmed for a set of automated tasks.

Bots are computer systems that are programmed for a set of automated tasks.

Just like their human counterparts, there are both good bots and bad bots. Good bots are created to scan the Internet and gather intelligence on websites. They are designed to perform repetitive tasks at a much higher rate than people. Many reputable companies, including well-known names such as Google and Microsoft, use bots to gather information from the expanses of the Internet to index their search engines.

Bad bots are created when a threat actor gains control of either personal or corporate PC laptops, servers, or mobile devices such as smartphones and tablets, or by creating new cloud instances. In the latter case, many new bots are created from free trials of cloud accounts. This is low risk and easily automatable, allowing the attacker to utilize free resources on cloud systems with a huge amount of connectivity and processing power behind them. This newer category is referred to as a BotCloud or CloudBot.

Why Bad Bots Are Created

The business of creating and controlling bots is called "bot herding." A group of bots is called a botnet. Hackers create botnets to achieve various objectives, the most common of which is to make money. Some bot herders leverage bots for their own nefarious activities, but most create them to lease or sell access to other hackers or hacking groups. Those wishing to rent bots can lease them by the hundreds to the thousands for anywhere between two cents and fifty cents each, depending on the volume of bots desired or the bot's geo-location, processing power, connection bandwidth, and various other factors.

What Bad Bots Do

Bots and botnets have a broad range of uses including mass reconnaissance, distributed denial of service (DDoS) attacks, beachheads in a target environment, application attacks, ad click fraud, account takeover, attacking application programming interfaces (APIs) for web apps, and various forms of financial fraud. Because they are not associated with the attacker, automatable, distributed globally, and ultimately disposable, bots are especially effective at many of these objectives. If the bot is detected or blocked, the attacker faces no personal risk. If even a single bot is successful at obtaining the objective, that single success can pay for the fees and pay dividends on top.

Why Bad Bots Are A Significant Threat

The Expansion Of Bad Bots

Bad bots generate about 19% of the total Internet traffic globally.[1] This is a bit disturbing, considering at any given time most bad bots are lying dormant until activated by their owner. Researchers also estimate that as many as 15% of Internet-connected devices, about 24 million, were compromised and tasked as a bot, a situation known as "botted." Based on the change in volume year over year, bots are estimated to continue expanding by about five million per year. This growth far outweighs the decommissioning rate.

Bad Bots Exploit More Than Web Vulnerabilities

Though web vulnerabilities were monitored and tracked for years, it was not until 2015 that the Open Web Application Security Project (OWASP) released a list of automated threats exploiting the business processes of web applications, including scraping, carding, and credential stuffing. These are very different from technical web security flaws such as SQL Injection and Cross-Site Scripting, which topped multiple security flaw lists for over a decade. In contrast to the OWASP Top 10, OWASP Automated Threats are not coding or implementing other technical flaws in the programming of the business application, but rather they exploit and/or misuse inherently valid functionality within the application to achieve fraud. As such, they require different sets of tooling for defense.

The twenty threats listed below are focused on what bots achieve when exploiting the business logic of websites. The future of cybercrime focuses on exploiting these sorts of attacks to increase the revenue from web-related fraud. Thus, the future of web security is having solutions that solve the problems posed by the OWASP Automated Threats, not just the Top 10 vulnerabilities in the code.

OAT-001 Carding	**OAT-002** Token Cracking	**OAT-003** Ad Fraud	**OAT-004** Fingerprinting
OAT-005 Scalping	**OAT-006** Expediting	**OAT-007** Credential Cracking	**OAT-008** Credential Stuffing
OAT-009 CAPTCHA Bypass	**OAT-010** Card Cracking	**OAT-011** Scraping	**OAT-012** Cashing Out
OAT-013 Sniping	**OAT-014** Vulnerability Scanning	**OAT-015** Denial of Service	**OAT-016** Skewing
OAT-017 Spamming	**OAT-018** Footprinting	**OAT-019** Account Creation	**OAT-020** Account Aggregation

(You can find more information about these issues on the OWASP Automated Threats site)

Bad Bots, Automated Threats, And Cybercrime

Between 2013 and 2015, global cybercrime-related losses quadrupled. In 2015, Lloyds of London estimated that cybercrime-related losses were as much as $400 billion US. Losses are expected to more than quadruple again by 2019, reaching as much as $2 trillion US.

Cyber-related crime is accelerating at this rate due to the ability to automate transactions. Crime groups no longer need to pay banks of people to send emails, make calls, or fill out web forms. Bots can do the job for less cost with higher overall throughput. Where no restrictions on transaction interactions are applied, bots can work hundreds of times faster than a person operating at maximum efficiency with no fatigue or need for breaks.

To add insult to injury, there are also legitimate companies, not considered criminal enterprises, which are embracing the use of bots to scrape websites for competitive information like prices and proprietary content. This grey market of competitive intelligence scraping, while not considered illegal in terms of cyber-theft, has wider business ramifications to a victim company. Its pricing may be undercut, the website SEO might be damaged, and ultimately, the company's online sales may drop.

[1] Bad Bot Report 2016 (Distil Networks)

Techniques For Defense: Common Misconceptions

Many organizations still seem to believe that older technology and approaches are effective as they are, or can adapt to fight a cutting edge bot problem. These technologies have not advanced sufficiently to protect against the quickly evolving bot threat. The two most common traditional defense methods are discussed below.

JavaScript + Third Party API-based Detection

With this approach, a web application owner will include JavaScript code on each sensitive page and, if desired, every page of their site. When a bot interrogates the page with the JavaScript, the JavaScript triggers. Simple bots cannot respond because they do not have JavaScript enabled. Additionally, background requests performed by the web application owner's JavaScript (such as AJAX requests) can lead to misidentification as these requests are unable to execute the interrogating JavaScript.

Sophisticated attackers can program the bot to interrogate Java to determine which scripts to execute and which to ignore.

Fundamentally, the biggest problem with these types of solutions is they are out-of-band and not inline. To block on the signals they provide, you must introduce latency by connecting with the out-of-band infrastructure, asking that system what it thinks, and then program your own action to take. This adds significant labor to your team and potentially introduces significant latency versus an inline solution that makes a judgement on traffic immediately.

Beyond that fundamental problem, it is important to realize that this detection method has several other drawbacks and limitations. First, it is generally only a detection method, not a blocking method, unless used in conjunction with other defenses. This detection method only works if someone executes the included JavaScript. If they do not execute it, or they spoof it, you need to build your own mechanisms to detect the lack of execution and block those bots. Second, it does not start until the bots are already interacting with your site and as stated before they require additional tools to provide automatic blocking capabilities. Third, JavaScript-based bot blocking solutions have access to your application logic and source code, so if compromised by an attacker, they may provide improved intelligence or access to the application. Fourth, these types of solutions place an additional burden for implementation and maintenance on the web development team. Fifth, sophisticated attackers can program the bot to interrogate JavaScript to determine which scripts to execute and which to ignore.

Web Application Firewalls (WAF)

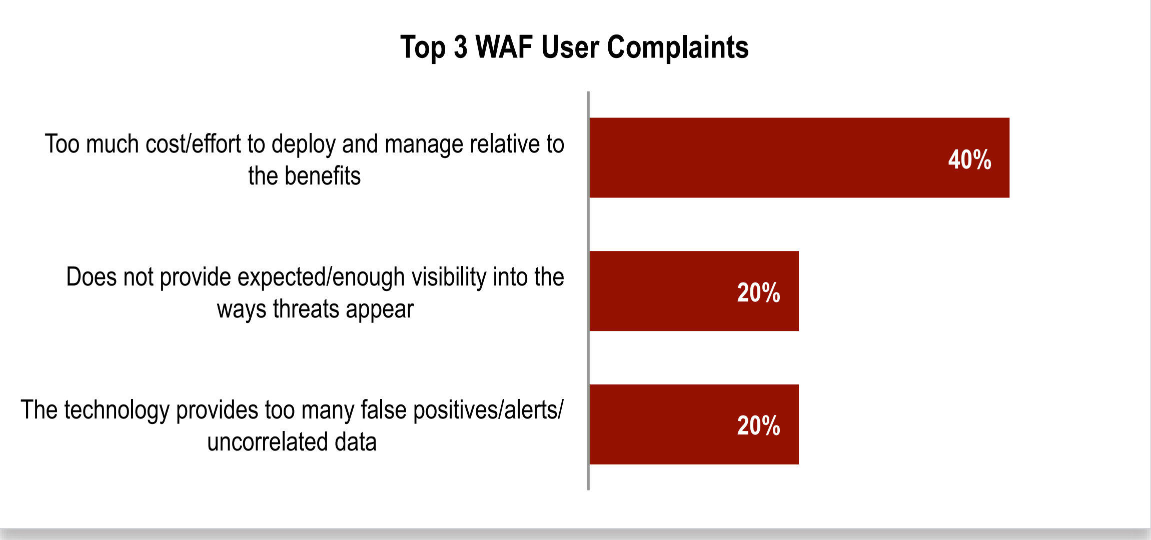
WAFs are appliances or software designed to inspect network communications from a client to an application to protect the application and backend database from exploitation. WAFs were created to have a higher degree of interrogation on the application communication flow than traditional application proxies and other firewalls in order to identify interactions with the application that could be fraudulent or malicious.

WAFs evaluate inputs from the clients looking for exploitation of application code, such as the OWASP Top 10 vulnerabilities, application coding flaws, system misconfigurations, and data leakage. WAFs generally identify these issues through the evaluation of "dangerous" strings submitted to the server or engineered packets and other communication-related abnormalities that match or violate signature-based rules or policies. Over time, they evolved by adding other capabilities like many of the features discussed previously, as well as client behavioral analysis (rates and patterns of website or application interrogation). The other features offer the same protections but are also susceptible to the same limitations. The idea is layered protection. The more detection methods are engaged, the fewer attacks will get through.

Good bots get along with WAF technology because they generally follow the rules of bot etiquette because their owners do not want to be blocked or banned from sites, while bad bots do not obey such rules. For the cases of bad bots, third-party evaluations place WAFs at 80% to 90% efficacy in detecting and/or stopping bots that attempt to exploit programming flaws such as the OWASP Top 10. However, they are also far less effective against bad bots attacking the OWASP automated threats list because they are not exploiting flaws in programming but business logic, which most WAFs do not sufficiently understand or protect against.

Respondents ranked WAF technology next to last for value on total cost of ownership out of 18 categories of technologies.

The biggest issue WAFs have is that although they can help stop some portion of bots, the reality is that bot detection was not what they were designed for. They are designed for application protection. Their rule- and policy-based approaches cannot adapt or scale to defend against large scale bot attacks. In its Data Driven Security Reloaded (DDSR) report, EMA asked WAF users about their key areas of dissatisfaction over their use of WAF technology. Forty percent (40%) of the respondents said their top problem was, “too much cost/effort to deploy and manage relative to benefits” (this was the top answer). Another of the top five answers was, “the technology provides too many false positives alerts.” Respondents were clear that their WAF implementations were expensive and labor intensive for the amount of benefit they received, supporting the statement that they have difficulty scaling and adapting. Many WAF solutions require regular tuning to keep the false positive alerts at a manageable level, thus making them labor intensive (in a previous role, the author of this paper spent 25% of his time each week tuning the WAF solution for his company and that was a product from an industry-leading vendor). DDSR respondents ranked WAF technology next to last for value based on total cost of ownership out of the 18 categories of technologies included in the research.



Techniques For Detecting Bot Fraud

With bot threats proliferating, web application owners need to understand the defenses available to protect their web properties and the issues with their efficiency. There are several methods administrators and security personnel traditionally use to identify bots and protect their properties. These methods all have limitations that make them less effective against advanced attacks.

Some of the oldest and most common methods of detection are log analysis, IP/Host/Domain Filtering, and CAPTCHA. These can be reviewed outside of this paper as this paper focuses on more recent methods and breakthroughs in detection and prevention.

Adaptive Defense Using Focused Bot Mitigation

The first thing to recognize is that bots are designed to exploit a broader range of issues than mainstream WAFs. Bots can attack user accounts and logins in numerous ways, perform advanced reconnaissance, create man-in-the-middle attacks, perform data mining, and exploit APIs, all of which are beyond normal application protection and therefore outside of the scope of WAFs.

Bots can attack user accounts and logins in numerous ways, perform advanced reconnaissance, create man-in-the-middle attacks, perform data mining, and exploit APIs, all of which are beyond normal application protection and therefore outside of the scope of WAFs.

Basic Fingerprinting For Bot Detection

Bot detection must start, but not end, with interrogating the initial client connection in real-time to determine if the client is a human or a bot. To do this, the solution may apply fingerprinting for the device and the user. This is done by submitting an escalating set of queries and challenges to the client to gather more detail about aspects of the host and user. Basic fingerprinting uses attributes like client IP address, hostname, and browser version. The key is that the basic fingerprints are limited in the data they collect and interpret, so if a bot spoofs any of these evaluation characteristics, it has a high probability of thwarting basic fingerprinting.

Advanced Fingerprinting For Bot Detection

For advanced fingerprinting, those changes do not invalidate the entire fingerprint since there is still enough other data gained from the client to "convict" it. Also, the more interaction the client has with the host, the more behavioral information is analyzed to further isolate it as an entity. For advanced fingerprinting, the key distinction is the assertive PULL of data from the client, as opposed to relying on headers that are pushed. This allows the gathering of specific data that helps improve identification and not relying on simply the data that the client wants to send. Advanced fingerprinting is really more like current facial recognition software. The subject can change aspects of its appearance to avoid recognition, but there is enough information in other data points to make a positive identification. When the fingerprinting capability adds increased detail via both data and metadata, it becomes far more accurate. Compounding this with "proof of work" or "proof of sentience" further increases reliability.

Machine Learning And Behavioral Analysis

Because bots have different levels of sophistication employed for attacks, the defensive system must also be able to adapt to known and unknown vectors. This requires the use of current advances in machine-learning algorithms. Using a combination of both supervised and unsupervised learning, the analysis engine can identify which clients are behaving differently from “normal” users that visit the site. Because detection can utilize unsupervised machine learning, the algorithms adapt as they gather more data from the daily traffic and thus become even more sensitive to threats.

Granular Controls

From the business’s perspective, it is imperative that real users are not terminated since termination causes a loss in monetization and negatively impacts loyalty to the product and brand. As the threats are identified, additional challenges can be presented to the user and/or risk scores can be increased or decreased based on the analysis and interactions. The site administrator can then set the tolerance level threshold and when that is met, more egregious violators can be readily terminated.

Community Sourcing For Accelerated Detection And Blocking

Once a bot is identified, the characteristic information that identified the bot is sent to the cloud and stored in a centralized, maintained, crowd-sourced database for all clients to use. If a bot shows up at another client location, it can be expeditiously identified and banned. A crucial part of “convicting” the bad bot is the scale of information that was collected about it during its interaction with the previous host or cumulative hosts.

By combining the community intelligence about each connecting host with the live fingerprint and behavioral characteristics, defenders create a comprehensive database of legitimate and malicious hosts that an advanced mitigation solution can use to track threats across the entire Internet.

Advanced API Security

APIs are a core part of many web applications today. They enable a direct interchange of information between two parties: server-to-server or client-to-server. Without them, many business processes would become at best cumbersome and in many cases infeasible, especially at the breakneck pace needed to support businesses today.

Bots exploit the very nature of APIs to extract as much data as possible in as little time as possible; in doing so they not only harvest data, they also significantly tax server resources, causing transaction delays, abandon purchases, and in the worst cases, system down time.

Bots exploit the very nature of APIs to scrape as much data as possible in as little time as possible.

Advanced security is designed to prevent content scraping of APIs and ensure API requests come from authorized sources, meaning only mobile users access the mobile API while web/app users access the web API, and they assist in governing API services to ensure licensed users do not abuse API.

Advanced API security leverages existing API authentication methods to enable quick deployment and lower maintenance. Dynamic rate limiting requires no additional developer time to integrate or manage. It provides great value by benchmarking normal API usage over time and then providing dynamic recommendations on the thresholds to use for client requests by time and by identity token.

Advanced API protection also addresses each environment or platform where the native application resides, providing an advanced fingerprint (e.g. facial recognition) for each device, which allows the administrator to restrict access to each native application environment.

System Challenges To Bots

Code Honeypots And Deception

When programming pages, coders can add honeypots or other programming-based traps that humans will not interact with. These triggers are embedded in the source code but are invisible to human clients. Bots identify and interact with these hidden triggers, thus exposing themselves. Once bots attempt to interact with hidden triggers, the defender can immediately counteract their interaction attempts.

The randomizing or manipulating of the source code means that the website is harder to program against, and this polymorphism makes the source code more difficult to predict and exploit because it changes dynamically.

Browser Validation Checking

Many bots impersonate different browsers but, when examined, do not match the attributes typically expected. Browser validation is handled in many ways and as bot sophistication increases, so must detection techniques. Every single browser that visits a site needs validation and this analysis must include a deep understanding of the differences between each version of a particular browser. As one example, best practice browser validation techniques check on the packet size of the header and the particular order of the header, and if these do not match with what is expected, then further validation in the form of challenges is initiated. These challenges vary from simple verification to determine if the browser is able to post back a JavaScript response, to the more complex, requiring the solving of a computational challenge.

In conjunction with browser validation checking techniques, bot mitigation solutions must also detect the use of sophisticated browser automation tools. These tools, like Selenium and PhantomJS, leave signatures that are detectable with the right expertise. To detect these "breadcrumbs," the defender must be able to perform an automated and rapid in-depth analysis of the underlying framework that supports these tools within the client or bot. Identifying inconsistencies in the header order, encoding methods, etc. and checking identifiers like user agent strings will flag bots interacting with the site.

Tamper Proofing

It is important to dynamically change the types of techniques you complete to avoid being exploited by spoofing or reverse engineering attempts from sophisticated threats. Tamper proofing techniques are more available with an inline solution. When you solely rely on a static piece of JavaScript in a webpage, then the malicious actor easily knows what to identify, evaluate, and defeat. By changing what you insert and how you insert it, and subsequently responding back with more challenges and tests, depending on the responses received, then you continuously keep verifying that your solution is not fooled.

Proof of Work algorithms issue a mathematical challenge to the browser through JavaScript. It is a relatively simple computation that a real client browser can process in the background without impact to the user. However, a fake browser from a bot will have difficulty providing a solution unless the programmer took the time to add this functionality to their bot build.

The Proof of Work defense relies primarily on the Hashcash algorithm which uses a hash function in the same way that HMAC or RSA signatures are defined on a pluggable hash-function. The key element behind the Proof of Work is the additional overhead it tasks the bot with. The additional problem solving requirements are often more than a bot can handle and thus cause the bot to fail at this method of interrogation.

While humans do not directly interact with cookies, bad bots can and often will. Advanced detection should also include an anti-tamper layer that identifies and even prevents bad bots from harvesting cookies, inserting fake cookies, or tampering with existing cookies.

EMA Perspective

Leveraging a layered defense is a prudent choice. That being said, identifying a single solution that can leverage as many of the defenses as possible is the only way to go. Having separate solutions for each defense is cost prohibitive and too resource-intensive to maintain.

Identifying a single solution that can leverage as many of the defenses as possible is the only way to go.

Additionally, detection cannot merely rely on a combination of static and/or reactive methods to stop both programming and business logic attacks; threat actors adapt to static defenses by creating new attack vectors and countermeasures. They are constantly evolving and adapting their approaches because to make money, they must be successful at overcoming defenses.

To combat this evolving threat, defenders must adopt a multi-layered, self-tuning defensive posture to move from the reactive to the proactive realm. With an array of adaptive controls in place, defenders can detect and prevent both humans and bot threat actors attempting to defraud their web properties.

A focused bot mitigation solution must incorporate advanced fingerprinting, community-sourced threat intelligence, advanced API protection, and system challenges to reliably verify identification of automated bot traffic.

About Enterprise Management Associates, Inc.

Founded in 1996, Enterprise Management Associates (EMA) is a leading industry analyst firm that provides deep insight across the full spectrum of IT and data management technologies. EMA analysts leverage a unique combination of practical experience, insight into industry best practices, and in-depth knowledge of current and planned vendor solutions to help EMA's clients achieve their goals. Learn more about EMA research, analysis, and consulting services for enterprise line of business users, IT professionals and IT vendors at www.enterprisemanagement.com or blogs.enterprisemanagement.com. You can also follow EMA on Twitter, Facebook or LinkedIn.

This report in whole or in part may not be duplicated, reproduced, stored in a retrieval system or retransmitted without prior written permission of Enterprise Management Associates, Inc. All opinions and estimates herein constitute our judgement as of this date and are subject to change without notice. Product names mentioned herein may be trademarks and/or registered trademarks of their respective companies. "EMA" and "Enterprise Management Associates" are trademarks of Enterprise Management Associates, Inc. in the United States and other countries.

Corporate Headquarters:
1995 North 57th Court, Suite 120
Boulder, CO 80301
Phone: +1 303.543.9500
Fax: +1 303.543.7687
www.enterprisemanagement.com
3438.101416