bugcrowd

# PRIORITY ONE

## The State of Crowdsourced Security in 2019

# Table of Contents



## Intro: State of Security in 2019

The global security threat outlook evolves with each new year. New technological innovation brings new attack vectors, increasing the number of ways that known and unknown vulnerabilities can be exploited, potentially causing damage to everyone from large enterprises to small businesses and individuals.

It is clear that there is no shortage of vulnerabilities to find. In the last year, Bugcrowd saw a 92% increase in total vulnerabilities reported over the previous year. The average payout per vulnerability increased this year by a whopping 83%, with average payouts for critical vulnerabilities reaching $2,669.92 — a 27% increase over last year.

Over the last few years we saw mega-bugs like ETERNALBLUE, Double Kill, Meltdown, Spectre, and the vulnerability in Apache Struts2 — which was responsible for the Equifax breach — just a few examples of bugs that were exploited in ways that made headlines and left many systems, users and companies devastated.

It's not all bad news though: What doesn't always hit the headlines are the breaches that never were — the stories of companies who "hacked themselves first" and drove up the cost of exploiting vulnerabilities in their systems by rewarding ethical hackers to identify and disclose these flaws to them, enabling these organizations to fix them before they could be exploited.

For CISOs, it can be hard to measure the risk, money, time, and reputational cost of a breach, but as the crowdsourced security market matures we're starting to see some consistent themes emerging from our customer's success stories:

Last year, our Crowd — the community of hackers, researchers, and pentesters on the Bugcrowd's Crowdcontrol platform — identified the same Struts2 vulnerability that attackers exploited to breach Equifax for one of our Fortune 500 financial services customers. As a result, this global company reduced its exposure window from months to days, and averted a similar outcome at a tiny fraction of the $1.4B Equifax ending up paying.

Richard Rushing, CISO of Motorola Mobility, also put a price on it:

**"Bugcrowd saved us $60 million from a single vuln, simply because we've avoided major data breaches in the eyes of our customers."**

These are just a couple of the real-world examples of the ROI that comes from harnessing the power of the Crowd.

## Methodology

Bugcrowd's Priority One Report analyzes proprietary platform data collected from thousands of crowdsourced security programs, many hundreds of thousands of vulnerability submissions, and tens of thousands of hours applied by the Bugcrowd bounty management teams since 2012. The report is a holistic, data-driven, and industry-segmented look at crowdsourcing adoption, its economics, the whitehat hacker community, and the vulnerabilities.



## Key Findings

Total payouts increased 83% year over year.

The average payout for a critical vulnerability in 2019 is $2,669.92, a 27% increase year over year.

In the first half of 2019, we saw a 29% increase in the number of programs launched versus the same time the year before and a 50% increase in public programs launched

> Why: More companies are reaching security maturity and taking their programs public as a part of their corporate social responsibility on the Internet.

Submissions have increased 92% overall, with submissions on IoT targets increasing more than any other at 384%.

> Why: More IoT targets + more security researchers specializing in IoT = more submissions. Unfortunately, the issue of systemic vulnerability in the IoT space is still a very real problem.

In line with this, payouts on IoT targets were second highest, following payouts on web which remained highest.

# Priority One: Critical Bugs

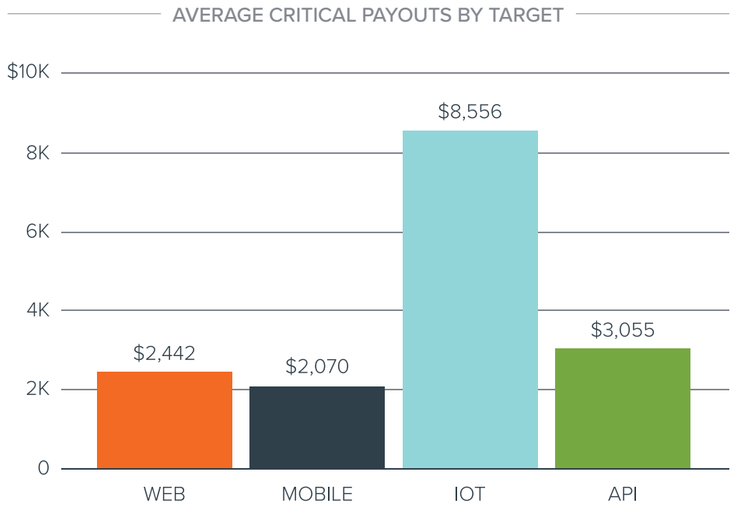
There was a major shift in vulnerability classes found by security researchers in 2018. Nearly 90% of the critical and high vulnerabilities found in 2017 were classes that we have traditionally seen year in and year out. In 2018, the focus shifted away from easier-to-find bugs and low hanging fruit.

## Why was that?

Looking at the data, 4 out of 5 of the top VRT classes for 2018 revolve around vulnerabilities that are difficult, if not impossible for any machine to find. Broken Access Controls, Sensitive Data Exposure, Server Security Misconfiguration, and Broken Authentication & Session Management are systemic issues with critical impact, and very few programming frameworks out there that protect against them. The ones that do are far from perfect.

**TOP 5 VULNERABILITIES OVER THE LAST YEAR**

1. Broken Access Control
2. Sensitive Data Exposure
3. Server Security Misconfiguration
4. Broken Authentication & Session Management
5. Cross-Site Scripting




# Top Vulnerabilities of 2018-2019 (OWASP Top 10)

Given the volume of web and API properties Bugcrowd works with, it's no surprise that the top 5 vulnerabilities our Crowd discovered over the past year are all on the OWASP Top 10 list. Big vulnerabilities elicit big headline grabbing breaches that affect millions of consumers. In reality, the vulnerabilities that lead to these big breaches are often much more nascent. The big bugs we see most often on our platform are also the most common. What's more interesting is the way our Crowd of whitehat hackers works to identify these vulnerabilities, chain them together, and create a greater impact through different methodologies and attack scenarios.

While the vulnerabilities in IoT devices – refrigerators and DVRs – capture our attention for their novelty and fear factor, they are still and by far outnumbered by vulnerabilities in web applications. In fact, web application vulnerabilities have always been the top submitted vulnerabilities across our programs and correspondingly account for the highest percentage of awards paid.

But what are the top submitted vulnerabilities on web applications? On the next page is a list of the top vulnerabilities we've seen, by volume, over the last 12 months.

1

Cross-Site Scripting (XSS) ➜ Reflected ➜ Non-Self

2

Cross-Site Scripting (XSS) ➜ Stored ➜ Non-Privileged User to Anyone

3

Broken Access Control (BAC) ➜ Insecure Direct Object References (IDOR)

4

Broken Authentication & Session Management ➜ Privilege Escalation

5

Unvalidated Redirects & Forwards ➜ Open Redirect ➜ GET-Based

6

Cross-Site Request Forgery (CSRF) ➜ Action-Specific ➜ Authenticated Action

7

Broken Authentication & Session Management ➜ Failure to Invalidate Session ➜ On Password Reset and/or Change

8

Server Security Misconfiguration ➜ Mail Server Misconfiguration ➜ Email Spoofing on Email Domain

9

Broken Authentication & Session Management ➜ Authentication Bypass

10

Broken Authentication & Session Management ➜ Failure to Invalidate Session ➜ On Logout (Client & Server-Side)

While it likely won't surprise most that cross site scripting (XSS) remains number one, it's important for understanding the threat landscape. Three of the top ten bugs (Access Controls and Authentication related) are predominantly classified as P1, the most critical submission on Bugcrowd.

If you step away from sheer volume and begin to look at criticality, the Bugcrowd data for critical P1 and P2 vulnerabilities yields other more insidious bug classes as well:

Server Side Request Forgery

Insecure Direct Object Reference

XML External Entity Injection

Security Misconfigurations in:

- AWS S3 buckets
- Git repo leakage
- Subdomain takeovers
- API keys and default credential exposure

Variants of command and code injection (RCE)

Blind XSS/Stored XSS & XSS + CSRF chains leading to full user account takeover

Understanding the most common vulnerabilities is important for the defenders who continuously face the challenge of making remediation decisions around vulnerabilities without access to all of the facts, and a key point of learning for bug hunters, especially those who are just getting started with bug hunting. It's also why the **VRT (Vulnerability Rating Taxonomy)** is so important for quickly determining business impact, and ensuring the expectations of both bug hunter and defender are kept aligned.

## CRITICAL BUGS: NOW & IN THE FUTURE

Researchers finding critical bugs are no longer going after things like XSS, CSRF, and SSI as those are fairly easy to find by many scanners out there today. This major shift is signaling a new era of deep testing like we've never seen before. Many organizations out there today have employed scanners and the like to protect against very easy to find, low hanging fruit. But where do you go from there once all of that automation is set up?

While penetration testing will still be done, how will that fit into your SDLC? Organizations are moving to much more agile methodologies to keep up with their development, but are still using the security equivalent of waterfall when it comes to relying on traditional penetration testing to help ensure their always-evolving environments are kept secure. It's fairly obvious that mixing the two together will inevitably introduce lapses in coverage and efficacy. With this in mind, customers are now moving towards more continuous models of crowdsourced pen testing with the Next Gen Pen Test to help keep up with the changing landscape.

### REPUTATIONAL COST OF A BUG: WHY MOTOROLA TURNED TO CROWDSOURCED SECURITY

Motorola Mobility is one of the world's largest consumer electronics and telecommunications companies. As part of its robust security program Motorola runs both private bug bounty and vulnerability disclosure programs. Motorola not only recognized the need to connect with the security researcher community to find critical vulnerabilities quicker and more efficiently, they also recognized the reputational cost of a vulnerability found by the wrong crowd.

**"With all the security technology and process that we have in place at Motorola we always find bugs when product goes live. With one critical submission, Bugcrowd saved us close to $60 million, simply because we've avoided major data breaches in the eyes of our customers."**

**Richard Rushing, CISO**
**CISO of Motorola Mobility**

Learn more about why Motorola turned to Bugcrowd **here.**

# State of Bug Bounty

Up and to the right. Adoption continues to climb with more companies across industries adopting crowdsourced security programs. What we coined as the “state of bug bounty” five years ago has become much broader. Crowdsourced pen testing and vulnerability disclosure are growing at breakneck pace and the number of companies running programs for multiple years has resulted in a marked increase in the number of public programs.

The market has matured a lot in the last seven years. The proof is in the stats below.

### ADOPTION (PROGRAMS BY INDUSTRY)

**42%** YoY across industries

**71%** increase in Financial Services

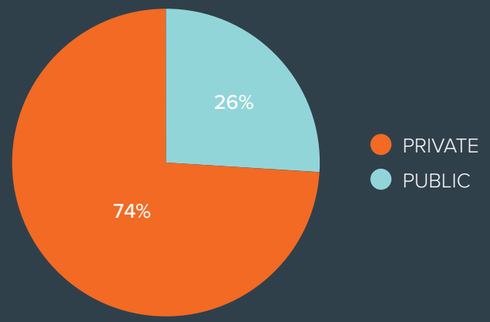
**50%** increase in Retail

**44%** increase in Healthcare

**26%** increase in Technology

**13%** increase in Automotive

### PROGRAMS RUNNING TODAY

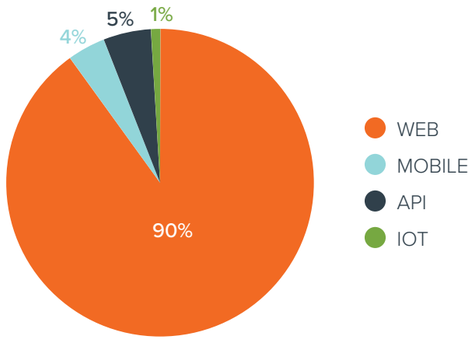




Over the last year, payouts on web targets remained the highest. It’s no surprise. The web is still the largest attack surface out there, and still accounts for 90% of all submissions. But others are gaining traction quickly. In the last year IoT payouts increased 384%, Mobile 141%, and API 101%. Year over year we saw an 83% increase in payouts — a result of increased, high priority submissions, as well as an increase in the number of programs with these targets in scope.

## 92% increase in total vulnerability submissions

### TOP TARGETS BY NUMBER OF SUBMISSIONS 2018




### SUBMISSIONS INCREASED BY TARGET 2018

- ↑ 99% Web
- ↑ 141% Mobile
- ↑ 101% API
- ↑ 384% IoT

### AVERAGE PAYOUTS ON CRITICAL (P1 + P2)

**27% increase**

### AVERAGE PAYOUT P1 – P4

**22% increase**

### TOTAL PAYOUTS YOY (2017 – 2018)

**83% increase**

# Getting Sh*t Fixed

We often refer to a vulnerability disclosure program as a neighborhood watch for the internet. Still, having a channel to receive vulnerability submissions from external researchers also requires a way to respond to these submissions. Below is a helpful checklist to help build out a response program.

1

The first step to proper vulnerability or incident response starts with taking all reports seriously and fully understanding the issue. Ignoring researcher submissions wastes time and leaves clients vulnerable longer. This isn't usually done maliciously; most often it's a case of the security team not fully understanding the magnitude of the report, or having adequate time to ensure everything gets the attention it deserves. It's not difficult to shift the narrative with a managed vulnerability disclosure program.

> In short, take all reports seriously until you're 100% clear on impact. There have been no shortage of findings that take a while to understand — be open and willing to have a conversation to completely understand what's being reported, and why.

2

After a critical issue has been identified, it's important to think beyond just filing a ticket and waiting for engineering to fix it. While it might be out of your immediate scope, it's now your responsibility to make sure the vulnerability gets remediated in a timely manner. Track down all the relevant parties, explain the risk, and escalate as needed. Critical findings should never get lost in the backlog, and security is no place for politics to endanger the trust of your end users.

> This is another valuable place where proper security training comes into play. If the entire organization is aware of the risk and is onboard with making security a priority, then it makes it a lot easier to get things fixed more quickly.

3

Thank and reward the reporting researcher. Communication here is key. Make sure they know they're valued. This is done by putting in the time, effort, and dollars. Be sure to tip if it's a particularly valuable finding!



4

Validate the fix. Often engineers may not fully understand what they're fixing, or why. Or maybe it was outsourced to someone who is three levels abstracted from where it was found, and they're just looking to make the POC go away. The fix may be partial (blacklisting one or two offending characters), uninformed, or ineffective altogether. Be sure the fix is sufficient, try to break it, review the code, and send it back if it's incomplete.

5

Once remediated, go back to the researcher and advise them as such — maybe they can find a way around that hasn't been considered by your team. Thank them again for their work, and let them know that you look forward to their future findings.

6

Increase your scope and rewards. Having avoided disaster, now more than ever it's important to double down on the reality that having a crowdsourced program can (and does) help you identify these issues before they're otherwise exploited in the wild. Make sure researchers are testing your full attack surface, and are heavily incentivized to do so.

7

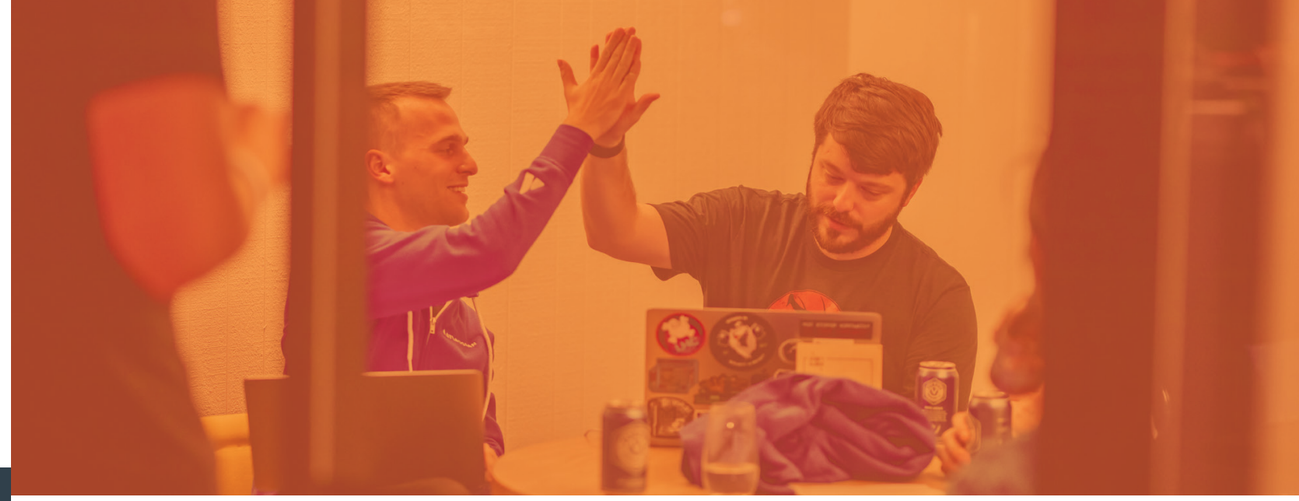
Rinse and repeat.

## INCENTIVIZING AN "OH SH*T" BUG IN FINTECH

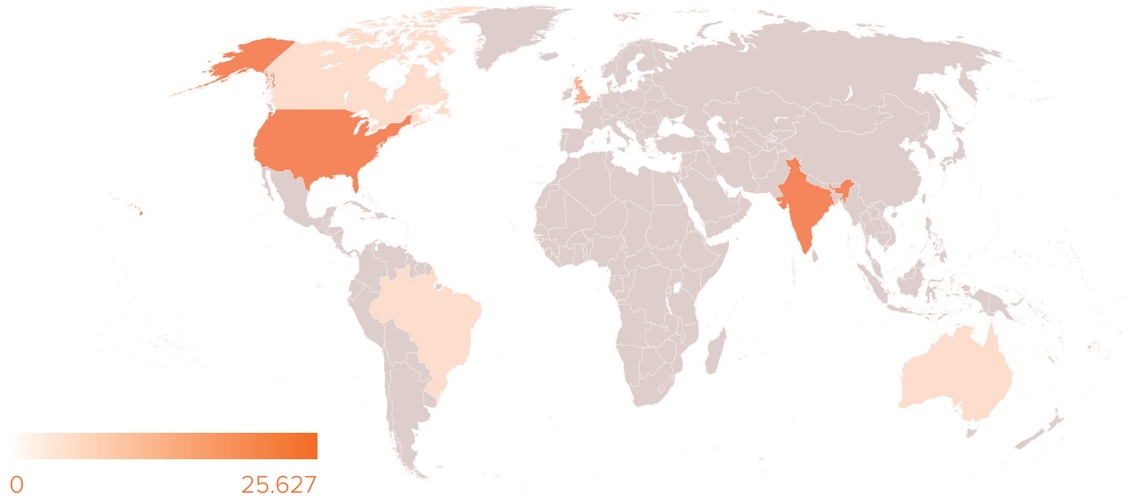
As a customer's program matures and their attack surface hardens, we often advise increasing scope and rewards. Once a process is established, and DevSecOps is in sync, finding and fixing bugs (or going from "oh sh*t" to fixed) becomes standard practice.

A recent example of this was a customer looking to put one of their applications to the test. To incentivize researchers to put in the time and effort, we recommended increasing rewards for the most critical bugs. High incentives, especially on a hardened application, ensure that if something is found, it's critical.

With this in mind, the customer increased their high reward to $20K — and it worked. Within two weeks, a researcher had identified a vulnerability that allowed takeover of user accounts. In the case of this app, this meant that a bad actor could effectively steal substantial amounts of money from other users. The customer quickly accepted, rewarded, and created a JIRA ticket for it, enabling them to fix a significant vulnerability before it impacted users.

## Snapshot: Inside the Mind of a Hacker




On top of customer rewards and bounty payouts, we have a set of additional incentive programs to keep our researcher engaged and excited to participate in Bugcrowd programs. In 2018 we paid more than $150,000 via our incentive programs to our Crowd. In addition to our MVP and Hall of Fame programs, in 2019 we've launched 2 new programs to reward our Crowd for all the amazing work they do: P1 Warriors and Bounty Slayers.

Both programs reward and recognize the valuable work our researchers do to make Bugcrowd's programs successful and help make the Internet safer for everyone! In addition, we modified our existing MVP and Hall of Fame programs to continue to challenge and encourage our researchers to reach new heights. As a result, we are projected to invest 50% more than last year on incentive programs for our Crowd.

## THE COMMUNITY THAT HUNTS TOGETHER

We recently asked some of Crowd how they got started and what advice they'd give others. The overwhelming response was to read blogs and follow the advice of other bug hunters. The idea that a hacker is someone sitting alone in front of a computer terminal is inaccurate. What we've learned over the last decade of working with this community is that it's all about giving back, about working together to make the internet a safer place. In fact, by collaborating, these whitehat hackers are able to chain together techniques to find more, critical vulnerabilities — ones they might not find on their own. Our **2019 Inside the Mind of a Hacker** report found that nearly 85% of security researchers are either already collaborating or looking to collaborate on bug bounty submissions in the future.

"Focus on basics and report quality. Read other researcher's blogs and bug bounty tips."

NIKHIL

"Use your time to read and sharpen your skills before thinking about rewards, there are a lot of blogs and sites that can help you get started. Believe in yourself and always ask questions to other people in the community that will guide you and help you to move along the right path."

MAJD ALDEEN ATIYAT (AKA TH3G3NT3LMAN)

The same idea applies to how they learn. Today, nearly **half of whitehat hackers** report that they learned how to hack via online resources. As such, it's important to engage with hackers and up-and-coming professionals in the way they prefer to learn. After all, these hackers will go on to use these skills to hack organizations ethically, identifying weak spots before the bad guys do.

Launched in 2017, **LevelUp** is our free online InfoSec conference series featuring leaders in the hacking and crowdsourced security space sharing their best practices, strategies, and research to help level up their fellow bug hunters.

Check out our videos from all of our **LevelUp** conferences on on **YouTube** and on **Bugcrowd University.**



### MENTORSHIP STORY: TH3G3NTL3MAN AND ISLAM

*We recently sat down with bug hunter, Islam (@SamExploit) about how he got started and how his mentor Majd Aldeen Atiyat (@th3g3ntl3man) helped him on his journey.*

**WHY DID YOU GET STARTED?**
My manager th3g3nt3lman kept telling me about bug bounty and what does it mean, how it changed his life and how it can change my life and others, so with this encouragement I said "why not?"

**HOW DID YOU GET STARTED?**
"After reading and watching a lot of videos about web penetration testing and bug bounty, I asked my mentor [if he wanted] to hack together twice a week so I could see how he thinks and how he approaches targets since he is a well known hacker and has a good reputation. He spent time explaining the Bugcrowd platform and how to use it, and explained how good they are and how much he likes the team, why he hacks mainly on their platform. By our third meeting I found multiple P1s and P2s and my life changed at that time."

**WHAT MOTIVATES YOU TO HELP OTHER BUG HUNTERS/WHY IS IT IMPORTANT?**
"When you feel the change that bug bounty has on your life — mentally and financially — you have to also help others who have the skills but don't know how to use them. The sweetness of finding a bug and getting rewarded for your efforts is something beautiful."

**WHAT ADVICE WOULD YOU GIVE TO OTHERS GETTING STARTED?**
"You have to spend time working on your skills and reading--there are a lot of online resources. Spend time with experienced and wise people who can guide you to the right path, and believe in yourself: you can do it."

*When asked why Majd Aldeen Atiyat (aka th3g3nt3lman) mentors, he said:*
"It's a responsibility, giving back to the community is a must, even though I had experience, I was junior in that area. A lot of amazing guys helped, guided and encouraged me to be the person I am today, so I have to share this experience with others and let them know bug bounty is a life changer."

### MENTORS

"I happen to be inspired by Eric (@todayisnew) because of his dedication and motivation to pentest or to help companies secure for his family."

"I was once a beginner and I find it very hard to start when you don't have any resources or someone who can help and guide you — that is why I decided to be part of the journey of many aspiring bug hunters as I can."

KENT BAYRON, BUG HUNTER

### DISCLOSE.IO/SAFE HARBOR

In 2018, Bugcrowd launched Disclose.io, an open-source safe harbor project. Over the past year nearly 100 companies have adopted this language in their crowdsourced security programs on Bugcrowd. To help move this along Bugcrowd has turned safe harbor on as a default for all new programs launching.

In early 2019, Bugcrowd rebooted "**The List**," a community-powered disclosure directory including bug bounty and vulnerability disclosure programs across the web that a) invite good-faith hackers to help and b) go the extra mile by offering safe harbor. In the short time The List has been published in its new form, we have had over 100 additions by the community, and several contributions and discussions on usability, scope, schema, and corrections.

The adoption of safe harbor by industry leaders is paramount to the success of crowdsourced security, and for the future of hackers and organizations working together to make our digitally-connected world safer.

Our goal is to have every organization offer a proactive vulnerability disclosure policy, and for that policy to include safe harbor for good-faith hackers. Here's what you can do:

### SEE A PROGRAM MISSING FROM THE LIST?

**Submit PRs** to include missing programs on The List – get them the praise they deserve and get credit for helping this movement out

If your organization runs a vulnerability disclosure program, consider adding the disclose.io safe harbor terms.

If your organization doesn't run a vulnerability disclosure program, talk to them about getting a policy, an intake channel, and a vulnerability coordination/remediation process established. **#ittakesacrowd**

## What's Changed Since Last Year

One thing that hasn't changed this year is the prevalence of web vulnerabilities, which are still on the rise. This trend will continue over the next year, although these vulnerabilities might come in different flavors as widespread migration to the cloud and mass adoption of IoT devices. Despite years of predicting this trend, security is still lagging behind in this area as it has not been built in at the core. For this reason, we'll have to go backwards, looking for "easy" bugs, especially in these newer computing environments. For the next few years we'll likely see a rise in vulnerabilities identified in these environments. This past year we hosted a car hacking bug bash  in Louisville, which perfectly highlights this trend. In just two days Bugcrowd security researchers, hackers and pen testers found over 15 critical and high severity vulnerabilities on vehicles.

Managing assets is also going to come to a head in 2019. This is a basic and fundamental issue that application security professionals continue to grapple with year after year. The bigger the organization and the more companies it acquires, the harder it is to manage assets. Most bugs are not found in flagship applications, but in obscure domains, apps, etc, that have been left behind and unaccounted for. Large Fortune 500 companies have a really hard time with this because they just don't know what they have on the internet. According to **Enterprise Strategy Group**, Large Enterprises have nearly 1,500 apps in production on average, but only 58% of those are protected. Even for smaller companies, lots of organizations have a hard time knowing exactly what they own – old internal apps, APIs, etc. The more we move to the cloud, the harder it is to track.

Moving to new technology environments is going to require more skill and education to combat the new vulnerabilities that may appear, as well as increased crowdsourcing to keep pace with the growing attack vectors. The cybersecurity skills shortage is growing at an alarming rate. In the years ahead we'll need to double down on recruitment and education, building security community and encouraging diversity. Bugcrowd University was created to teach basics of hacking and bug bounty hunting to address the skills shortage by introducing new researchers to the crowdsourced security field and upleveling the skills of the whitehat hacker community across the board. And this community continues to grow, and find critical bugs. In the last year our Crowd found 92% more vulnerabilities than the year before. With an 83% increase in payouts, it's clear that more, critical bugs are being found by a growing, engaged community of security researchers, whitehat hackers and pen testers.

We're going to see more crowdsourcing. With the model proven successful, we'll see more competitors crop up, and we will see new inroads into different crowdsourced security applications like forensics, threat hunting, and more. Next year is going to be about the individual contributors and tracking skill sets. We will eventually get to a point where a security professional doesn't have to take a consultancy job anymore. They can work from anywhere.

# Appendix

Here's an overview of the OWASP Top 10, why these vulnerabilities made the list and why many of them remain on the list, despite being known for decades.

## INJECTION

Among the oldest and most dangerous attacks aimed at web applications, injection vulnerabilities directly access stored data and can lead to data theft, data loss, loss of data integrity, denial of service, as well as full system compromise, among others. Listed as the number one risk in the OWASP Top 10, these dangerous vulnerabilities are typically caused by insufficient user input validation — i.e. Data from the outside of the application isn't checked and "sanitized" on it's way in to the application, leading to unintended consequences.

Injection vulnerabilities such as SQL Injection (SQLi) remain in the top spots due to their prevalence in legacy applications. The enormity of the attack surface, ease of introduction of these vulnerabilities into codebases, and the amount of time these vulnerabilities have been around makes them common targets for veteran attackers and newbies alike.

There is good news in this as well. The commonality of these vulnerabilities means that they are easy for defenders to identify.

### SQL INJECTION: THE GRANDFATHER OF WEB VULNERABILITIES

One of the most dangerous vulnerabilities in the web is also one of the oldest. First discovered by Jeff Forristal (aka Rain Forest Puppy) in 1998, SQL injection has remained one of the top security vulnerabilities since the 90s when most websites were still using simple Microsoft Access-based databases — and no real security properties protecting them.

## XSS

Like SQL Injection, Cross-site Scripting has made every OWASP Top 10 since the list's inception. Number 4 on the list, XSS is similar to SQL and injection vulnerabilities in that the core issue is the sanitization of data In this case the vulnerability is caused by the data going out of the application, as opposed to the data going in. A cross-site scripting vulnerability may be used by attackers to inject malicious client-side scripts that exploit the user's trust in the vulnerable domain, and allow for an attacker to execute a number of attacks against the victim including session hijacking, re-writing page content, redirection, and in some cases (particularly when chained with other misconfigurations) can lead to a complete account takeover.

The most famous version of XSS, and a solid explainer, is the **SAMY Myspace worm** by Samy Kamkar. This wormified exploit leveraged a stored XSS to trigger actions in the visitors browser when they visited an infected Myspace profile, ultimately leading to millions of Myspace users proclaiming that "SAMY IS MY HERO" over a 12-hour period.

The most common types of XSS attacks are stored (i.e. persistent) and reflected (i.e. transient). In a persistent XSS attack the injected script is permanently stored on the target servers, such as in a message forum, visitor log, username/address field, or comment box — in the example of the SAMY worm, the payload was stored in the comments of a Myspace profile.

In a reflected attack the injected script, usually via a query string parameter, is returned by the application and executed within the context of the victim's browser — such as in an error message, search result, or any other response that includes some or all of the input sent to the server as part of the request. This allows the attacker to execute arbitrary javascript on behalf of the victim.

## BROKEN AUTHENTICATION

Ranked number two on OWASP's 2017 Top 10: Broken Authentication and Session Management. Essentially if authentication and session management are not implemented correctly, attackers can compromise passwords, keys, or session tokens. In other words, attackers look for vulnerabilities in the authentication or session management functions to impersonate users or otherwise gain unauthorized access. As common as they are difficult to spot, broken authentication can result in full account takeover, privilege escalation, authentication bypasses, and other session related compromises.

Session management is a notoriously difficult problem for most organizations — especially those that store Personally Identifiable Information (PII), Protected Health Information (PHI), or even Payment Card Industry (PCI) data. Any compromise of data could spell disaster as seen with recent data breaches and could have long lasting negative effects on the brand. Let's not forget that GDPR will have even larger effects on organizations operating in the EU.

## SECURITY MISCONFIGURATION

Another big bug category that tends to hit hard, especially as organizations migrate to the cloud, are security misconfigurations. We see a lot of incorrectly configured cloud environments, such as in AWS and Azure—with clients leaking data due to infrastructure misconfigurations and incorrect implementations around data storage and setup. Another instance of this class is also if source code is managed in Github or other SVN/source repositories. With some easily misconfigured permissions, virtually anyone on the internet could potentially have access to their repositories and sensitive data.

Number 6 on OWASP's 2017 Top 10, Security Misconfigurations are incredibly common, dangerous, and at this point responsible for a disproportionate number of breached records in the form of S3 buckets, publicly accessible database backups, and other similar issues. Improper configurations lead to a number of issues, giving attackers unauthorized access to system data or functionality, and sometimes resulting in complete system compromise. One of the biggest culprits of this flaw is failing to change default passwords. There is a very easy fix for this potentially very big problem: turn everything off by default. Disable admin interfaces, debugging, and use of default accounts and passwords.

When you get into cloud, it's a whole new domain of technology, learning AWS and the associated technology stack is like learning a new language, you need a Rosetta stone for it sometimes. You need to be dedicated to it, and if you don't spend the time, it's easy to misstep. As we continue to move to new technology environments, we'll continue to see these bugs. These are the easiest vulnerabilities to find that have the largest impact on organizations. Having a healthy asset management plan is key to keeping this under control.

## SENSITIVE DATA EXPOSURE

Number 3 in OWASP's 2017 Top 10 is Sensitive Data Exposure. Sensitive Data Exposure is exactly what it sounds like: exposure of sensitive data such as banking, health, PII (social security number, data of birth) or user accounts, email addresses or passwords. How is this data exposed? There are a few ways: lack of encryption, failure to prevent browser caching, or even a forgotten or mistaken data upload. Even if the data is encrypted, weak keys or password hashing techniques can still give attackers access--and access to this data usually requires a manual attack: man-in-the-middle, stolen keys, or directly from a server while in transit.

What can consumers do? While most consumers will not know how to identify these issues, there are things to look for, such as whether or not the vendor has a vulnerability disclosure program. Given the frequency that this type of issue is found in our programs, it's likely that if a vendor is running a crowdsourced security program the issue has already been found...and fixed.

On a practical level, there are a few things consumers can do the keep themselves secure. The first is two-factor authentication (2FA). Most services (including ecommerce sites) support 2FA. Activate it wherever you can, especially on your personal email and social media accounts (i.e. the ones which, if accessed, can be used to reset all your other passwords and gain access to your accounts).

Minimizing password reuse on important services such as financial accounts (and those with critical PII data) is an important step. A password manager is the perfect tool for this. All of these tools such as 1Password, LastPass, and Keeper Security) have password reuse detection tools. Take advantage of these and give your internet identity review.



## INSECURE DESERIALIZATION

Number 8 on the OWASP Top 10 is Insecure Deserialization. Occurring when untrusted data is used to abuse the logic of an application or inflict a denial of service (DoS) attack.

Serialization operations are extremely common in architectures that include APIs, microservices, and client-side MVC. When the data being serialized and deserialized is trusted (under the control of the system), there are no risks.

However, when the input can be modified by the user, the result is an untrusted deserialization vulnerability.

The Equifax breach serves as a very real world example of what can happen when attackers discover insecure deserialization in the wild -- and why it's important to patch early and often.

### EQUIFAX

One of the most widely discussed breaches in the last couple of years was Equifax. The cause of this attack was exploited a vulnerability in Apache Struts — the result of unsafe deserialization in Java. The vulnerability enabled attackers to inject malicious code into any server running a Struts application that uses the popular REST communication method, and execute it — exactly what they did with Equifax.

## USING COMPONENTS WITH KNOWN VULNERABILITIES

Coming in at number 9 is Using Components with Known Vulnerabilities. While not a vulnerability in and of itself, this issue is both widespread and preventable -- and highlights the threats from using external dependencies in an application.

Most developers are focused on securing their own code and often forget about the code they have imported. In fact, they may not even know about all the code they are running — and the plugins, libraries and dependencies of that code.

What can you do? Make it your job to be aware of every system you're using and the security of those systems. Hold your vendors to a rigorous standard of security, but never assume they're aware of every vulnerability. At the same time, pay attention when they do issue patches and apply them quickly.

## INSUFFICIENT LOGGING & MONITORING

Number 10 on the OWASP Top 10, Insufficient Logging and Monitoring includes everything from unlogged events, logs that are not stored properly, and warnings where action is not taken in a reasonable amount of time. We all know the longer it takes to detect an attack or breach, the greater the potential impact. This is why this vulnerability is so critical. According to OWASP, "exploitation of insufficient logging and monitoring is the bedrock of nearly every major incident. Attackers rely on the lack of monitoring and timely response to achieve their goals without being detected."

Unlike some of the other vulnerabilities on the list, Insufficient Logging and Monitoring is directly tied to best practices. To protect your organization ensure login, access control failures, and server-side input validation failures can be logged with sufficient user context to identify suspicious or malicious accounts, and held for sufficient time to allow delayed forensic analysis. And don't forget to establish an incident response and recovery plan.

## XXE

XXE is a potent and often misunderstood vulnerability that, when weaponized, has the potential to result in outcomes as substantial as local file read, or even remote code execution. To exploit XXE an attacker sends a malformed XML payload, tricking the misconfigured XML parser to return internal or external resources — often containing sensitive information, such as local files on the machine (or again, in rare cases, code execution). For this reason it remains on the OWASP Top Ten at number 4.

XXE recently reared its ugly head in an Internet Explorer Zero Day that would have allowed attackers to steal files from the Windows systems if exploited.

### IS APACHE STRUTS SAFE?

Apache Struts has been in the news a lot over the last few years — most notably for the vulnerability that took down Equifax: CVE-2017-5638.

While CVE-2017-5638 has become synonymous with Equifax, this vulnerability had a much broader impact as many web applications out in the wild use Apache Struts. This is why vulnerability identification is so important. A researcher in our Crowd identified CVE-2017-5638 months before the Equifax breach on one of our customers who is a major worldwide financial services company. As a result, the customer remediated the vulnerability before a bad actor could take advantage of it. This customer did not end up on the news and as they say, no news is good news.



The simple answer is that nothing is ever 100% "safe" which is why ongoing vulnerability assessment is so critical. Awareness is key — not only for the vulnerabilities in your own code, but for those in the systems you use. Know what is in your environment and keep an eye out for issues. When patches are issued, update immediately.

As far as Apache Struts is concerned, there are public GitHub projects with ongoing research on both **CVE-2017-5638** and **CVE-2018-11776**, for full write-ups and gory details. Or, check out this proof of concept which walks through how the CVE-2018-11776 vulnerability **could be exploited.**

## BROKEN ACCESS CONTROL

In software applications, one bug that never seems to go away and one of the more risky vulnerabilities rated in our P1 or P2 VRT category, is the idea of access control bugs. These are really risky and there are a lot of different flavors of access control bugs. One of the riskiest is insecure direct object reference bugs.

Number 5 on the Top 10, OWASP defines insecure direct object reference bugs "a developer exposes a reference to an internal implementation object, such as a file, directory, database record, or key, as a URL or form parameter. An attacker can manipulate direct object references to access other objects without authorization, unless an access control check is in place."

For example, if someone has an account and that account has a unique number 75, and if a cyber attacker sees that in the URL and changes the number to 74, he or she can get access to someone else's account — that is an insecure direct object reference bug. Access control bugs are not easy to defensively code for. These bugs are not protected by any framework or code libraries like many input based vulnerabilities. As a result, we are going to continue to see this over the next year.