Hewlett Packard Enterprise

HPE Security Research

Cyber Risk Report 2016

Table of contents

Introduction

Welcome to the Hewlett Packard Enterprise (HPE) Cyber Risk Report 2016. In this report we provide a broad view of the 2015 threat landscape, ranging from industry-wide data to a focused look at different technologies, including open source, mobile, and the Internet of Things. The goal of this report is to provide security information leading to a better understanding of the threat landscape, and to provide resources that can aid in minimizing security risk.

About Hewlett Packard Enterprise Security Research

HPE Security Research conducts innovative research in multiple focus areas, delivering security intelligence across the portfolio of HPE security products. In addition, our published research provides vendor-agnostic insight and information freely to the public and private security ecosystems.

HPE Security Research brings together data and research to produce a detailed picture of both sides of the security coin—the state of the vulnerabilities and threats composing the attack surface, and the ways adversaries exploit those weaknesses to compromise targets. Our continuing analysis of threat actors and the methods they employ guides defenders to better assess risk and choose appropriate controls and protections.

HPE Security Research publishes detailed research and findings throughout the year, but our annual Risk Report stands apart from the day-to-day opportunities and crises our researchers and other security professionals face.

Just as HPE has evolved to stay ahead of the challenges brought on by growing frequency and sophistication in enterprise attacks, the threat landscape and how we protect the digital enterprise has also transformed. Taking a retrospective look at this changing landscape provides critical insights into the most prominent cyber risks while offering intelligence to enterprises looking to focus security investments and resources.

In 2015, we saw a continued rise in attackers' success at infiltrating enterprise networks, making it all the more critical for HPE's cybersecurity research team to provide this unique perspective on significant trends in the marketplace. Just as attackers continue to evolve their techniques, defenders must accelerate their approach to detection, protection, response, and recovery.

Our research saw an increased sophistication of attacks, even as the security world is encumbered by the same issues that have plagued us for years. The work done by our research team shows that even as regulations become more complex and attack surfaces continue to grow, foundational problems exist that challenge even the best defender. Our more sophisticated customers are responding to these threats, but many small and mid-market customers are not, thus making them an easier target.

Security practitioners from enterprises of all sizes must embrace the rapid transformation of IT and ready themselves for both a new wave of regulations and an increased complexity in attacks. The HPE Security Research group continues to prepare for the challenges—and the opportunities—the future will doubtless hold. It remains our fullest intention to invest in driving our thought leadership throughout the security community and to share our findings as they become available.

Sue Barsamian
Senior Vice President and General Manager
Hewlett Packard Enterprise Security Products

Our data

To provide a broad perspective on the nature of the attack surface, the report draws on data from HPE security teams, open source intelligence, ReversingLabs, and Sonatype.

Key themes

Theme #1: The year of collateral damage
If 2014 was the Year of the Breach, 2015 was the Year of Collateral Damage as certain attacks touched people who never dreamed they might be involved in a security breach. Both the United States Office of Personnel Management (OPM) and the Ashley Madison breaches affected those who never had direct contact with either entity, and whose information resided in their networks only as it related to someone else—or, in the case of the Ashley Madison breach, did not appear at all but could be easily deduced from revealed data. With the OPM breach, the true targets of the breach may be people who never themselves consented to inclusion in the OPM database—and who may be in danger thanks to its compromise. Data compromise is no longer just about getting payment card information. It's about getting the information capable of changing someone's life forever.

Theme #2: Overreaching regulations push research underground
When horrific events occur impacting the lives of many, there is a natural reaction to do something to try to prevent future occurrences. Too often, the "something" (legislation) incurs unwanted consequences to go along with the intended result. This is the case with various proposed regulations governing cybersecurity. While the intent to protect from attack is apparent, the result pushes legitimate security research underground and available only to those denizens who dwell there. To be effective, regulations impacting security must protect and encourage research that benefits everyone.

Theme #3: Moving from point fixes to broad impact solutions
While it is laudable that Microsoft® and Adobe® both released more patches than at any point in their history, it remains unclear if this level of patching is sustainable. It strains resources of both the vendor developing the patch and the customer deploying the patch. Microsoft has made some headway with defensive measures that prevent classes of attacks. It and others must invest in these broad, asymmetric fixes that knock out many vulnerabilities at once.

Theme #4: Political pressures attempt to decouple privacy and security efforts
A difficult and violent year on the global scene, combined with lingering distrust of American tech initiatives in the wake of revelations by Edward Snowden and other whistleblowers, led to a fraught year for data privacy, encryption, and surveillance worldwide. Many lawmakers in the US, UK, and elsewhere claimed that security was only possible if fundamental rights of privacy and due process were abridged—even as, ironically, the US saw the sunset of similar laws passed in the wake of the September 11, 2001, attacks. This is not the first time that legislators have agitated to abridge privacy rights in the name of "security" (more accurately, perceived safety), but in 2015 efforts to do so could easily be compared to the low success of previous efforts made after the attacks of 2001. Those evaluating the security of their enterprises would do well to monitor government efforts such as adding "backdoors" to encryption and other security tools.

Theme #5: The industry didn't learn anything about patching in 2015
The most exploited bug from 2014 happened to be the most exploited bug in 2015 as well—and it's now over five years old. While vendors continue to produce security remediations, it does little good if they are not installed by the end user. However, it's not that simple. Applying patches in an enterprise is not trivial and can be costly—especially when other problems occur as a result. The most common excuse given by those who disable automatic updates or fail to install patches is that patches break things. Software vendors must earn back the trust of users—their direct customers—to help restore faith in automatic updates.

Theme #6: Attackers have shifted their efforts to directly attack applications
The perimeter of your network is no longer where you think it is. With today's mobile devices and broad interconnectivity, the actual perimeter of your network is likely in your pocket right now. Attackers realize this as well and have shifted their focus from servers and operating systems directly to applications. They see this as the easiest route to accessing sensitive enterprise data and are doing everything they can to exploit it. Today's security practitioner must understand the risk of convenience and interconnectivity to adequately protect it.

Theme #7: The monetization of malware
Just as the marketplace has grown for vulnerabilities, malware in 2015 took on a new focus. In today's environment, malware needs to produce revenue, not just be disruptive. This has led to an increase in ATM-related malware, banking Trojans, and ransomware.

The business of bugs

In the modern world, having insecure software and services can negatively impact the bottom line of an enterprise. As breaches become more prevalent, the tools and techniques used in these breaches gain legitimacy and a monetary value. Put more simply, 2015 saw the culmination of the monetization of vulnerabilities.

Looking at the past year, it becomes clear that security researchers play an increasingly important role in identifying security vulnerabilities and investigating state-sponsored threats. As they do so, researchers become increasingly misunderstood, bordering on imperiled, by well-meaning governments resorting to broad legislation to protect themselves and their citizens from attacks. The playing field on which the information security community operates may have undergone a major shift in 2015, but it's been a long time coming.

Hacking Team exposes the gory details

Hacking Team, a Milan-based company selling offensive technology, found itself the victim of a breach resulting in the release of company emails, passwords, and documents.[1] This breach gave everyone a rare look into the inner workings of a zero-day exploits vendor. Hacking Team began moving from a traditional defensive information consultancy to a surveillance business in 2009 with the cultivation of relationships with zero-day vendors. It began purchasing exploit packs but was not impressed with the quality.[2]

In 2013 it made several new contacts and continued to grow external relationships with zero-day providers.[3] The exposure of detailed exploit deals and Hacking Team's customer list has allowed us to check assumptions about the zero-day marketplace, revealing pricing, exploit quality, and limiters to the surveillance business, such as the Wassenaar Arrangement.

20 years and counting

Incentivizing security researchers to find critical vulnerabilities in software has been a tactic employed by software vendors, security companies, and—more recently—B2B/B2C entities for two decades. Netscape initiated its rewards program in 1995 and is most commonly credited with establishing the concept of "bug bounties." Rewarding skilled researchers for identifying potential avenues to the enterprises' crown jewels has taken many forms, from public recognition to money, and everything in between. Over the past couple of years there has been growth in the number of organizations outside of high tech running bug-bounty programs. There has also been an increase in the number of third-party platforms (e.g., Bugcrowd, Crowdcurity, and HackerOne) as they manage the operational end of the program, saving their customers significant expense in doing so themselves. The Zero Day Initiative (ZDI), at the time a part of Hewlett Packard Enterprise (HPE), operated a hybrid version of a bug-bounty program, which accepted critical vulnerabilities in enterprise software including that offered by HPE.

Over the course of 2014 and 2015 there has been an observable increase in vendors launching programs—either through third-party platforms as mentioned above—or doing so themselves. Bugsheet, a community-curated list of bug-bounty and disclosure programs, is currently tracking more than 350 programs[4] with varying rewards (bounties, acknowledgements, or swag).[5] Bugcrowd also tracks a list of bug-bounty and disclosure programs as reported by their researcher community and lists over 450 programs, noting whether they pay a reward, give acknowledgements, or provide swag to the participating researchers.[6] Many companies appear on both lists. Historically, the vendors offering bug bounties have been in the high-tech industry, but we are starting to see a growth in non-IT industries joining in, especially as they look to breach risks related to their online presence.

Consistent security at scale is incredibly hard to achieve alone. Running a bug-bounty program expands a company's available resources more affordably. Some of the most notable programs in the past 20 years (Figure 1) can attest to this fact. Engaging the community returns many high value bugs the vendor may never learn about or only learn of through an attack in the wild (i.e., used in active attacks). None of the bug-bounty programs are about silencing the researcher. While some may view it as hush money, the research community sees it as payment for its work—especially when it reports a clever edge case.

[1] http://www.forbes.com/sites/thomasbrewster/2015/07/06/hacking-team-hacked/.

[2] https://tsyrklevich.net/2015/07/22/hacking-team-0day-market/.

[3] https://tsyrklevich.net/2015/07/22/hacking-team-0day-market/.

[4] http://bugsheet.com/directory.

[5] This list is only reporting vendor programs and not those run by third-party platforms.

[6] https://bugcrowd.com/list-of-bug-bounty-programs.

ZDI@10

One of the oldest bug-bounty programs around, the ZDI was founded in 2005 to protect the IT ecosystem by compensating independent researchers for submitting their finds to the program. Since that launch and since its purchase by Hewlett Packard in 2010, ZDI grew into the world's largest vendor-agnostic bug-bounty program. In the fall of 2015, Hewlett Packard Enterprise announced the signing of a definitive agreement to divest the TippingPoint business and ZDI to Trend Micro. Throughout the lifespan of the program, the core principles established at the beginning have remained in force (Figure 2).

The ZDI follows iDefense's middleman model. In fact, the ZDI was originally founded at TippingPoint by the same people that created iDefense's program.[7] In 2007, Dragos Ruiu started the Pwn2Own contest to run at his CanSecWest security conference.[8] The initial contest prize was a laptop, but later upgraded to a $10,000 reward provided by ZDI. Pwn2Own proved to be a great success and became a recurring event at CanSecWest. In 2012, the ZDI and Dragos teamed up to launch Mobile Pwn2Own (mPwn2Own) at the EUSec conference in Amsterdam. It later moved to Tokyo and the PacSec conference. In 2014, the event paid out $850,000 in rewards to skilled security researchers for more than 30 vulnerabilities, the highest contest payout to date.[9] In 10 years, the ZDI program has paid more than $12M and disclosed more than 2000 vulnerabilities, with another 300+ with vendors awaiting patch.

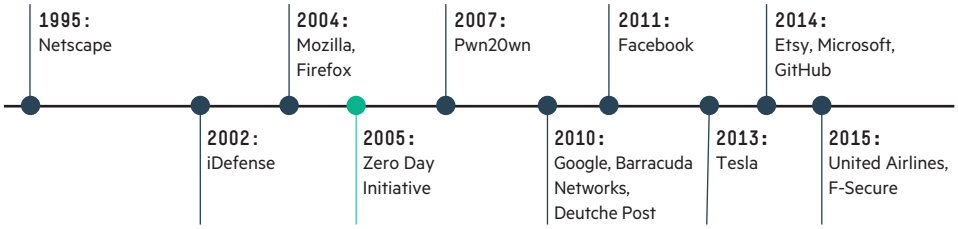


Figure 1. Timeline of notable bug-bounty programs, 1995-2015


Encourage the **reporting** of zero-day vulnerabilities responsibly to affected vendors


Fairly **compensate and credit** the participating researchers


Protect our customers and the broader ecosystem

Figure 2. ZDI core principles

[7] http://community.hpe.com/t5/Security-Research/HP-Zero-Day-Initiative-Life-begins-at-10/ba-p/6770464.

[8] http://seclists.org/dailydave/2007/q1/289.

[9] http://community.hpe.com/t5/Security-Research/HP-Zero-Day-Initiative-Life-begins-at-10/ba-p/6770464.

In 2010, Google™ entered the fray with a rewards program for Chromium followed by one for its web properties, thus launching the trend toward bug-bounty programs for web applications.[10] Over the years, Google has also co-sponsored Pwn2Own and mPwn2Own with the ZDI.

As attackers expand their focus to include nearly anything connected to the Internet, we see a corresponding response from non-software, non-IT companies entering the bug-bounty community. In 2013, Tesla Motors created a bug-bounty program, later expanding it and handing it over to Bugcrowd to manage.[11] Earlier this year, the first airline joined the community. United Airlines focuses on vulnerabilities reported against its websites, applications, and online portals and rewards researchers in a rather unique way—with 50,000 to one million air miles.[12]

While working to gain researcher interest and loyalty on the one hand, a successful bug-bounty program must also establish contacts with the affected vendors. Gaining their trust is crucial to long-term success. Not surprisingly, conversations between vendors and ZDI have been both congenial and contentious—often during the same conversation. Ultimately, most have come to trust that the program is helping them and our collective customers.

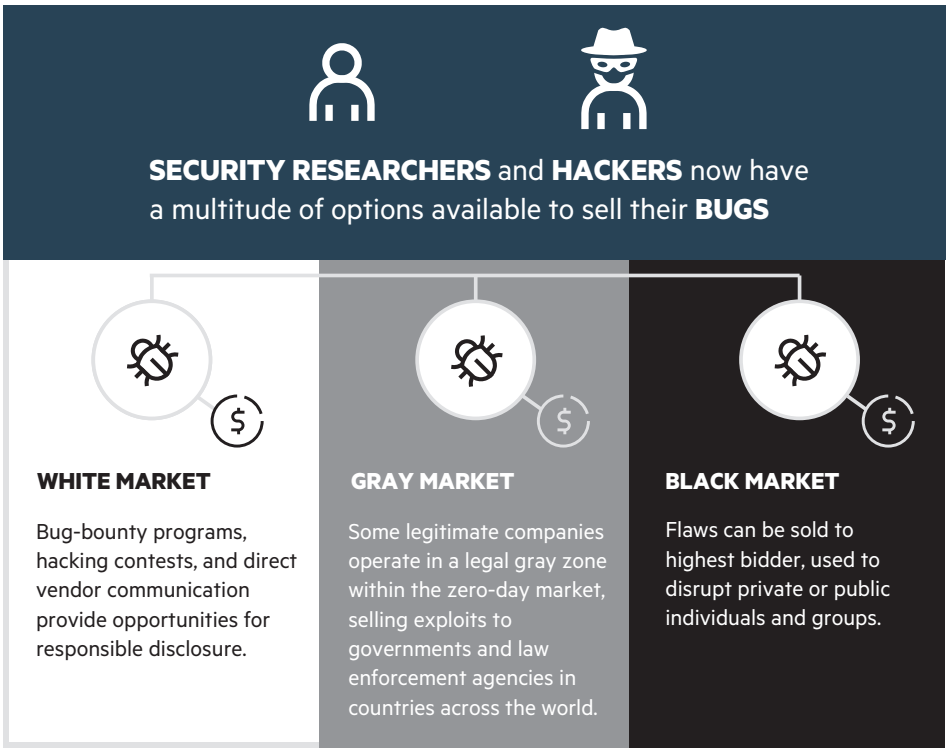


Figure 3. Vulnerability marketplace options

Different types of bounties
White/Gray/Black

At its core, a bug bounty is a cooperative relationship with the intent of identifying and correcting application vulnerabilities before they are exploited in the wild. Identifying application vulnerabilities has become a lucrative business with its own marketplace and players. When talking about the various players in this market, their ethics and motivations are often the first thing questioned, with categorization by the color of a hat (black, gray, or white) following close behind. The things that actually differentiate the players are the marketplace and the government under which each of them finds himself operating. Let's start with a look at the marketplace.

[10] https://cobalt.io/blog/the-history-of-bug-bounty-programs.

[11] https://bugcrowd.com/tesla.

[12] https://www.united.com/web/en-US/content/Contact/bugbounty.aspx.

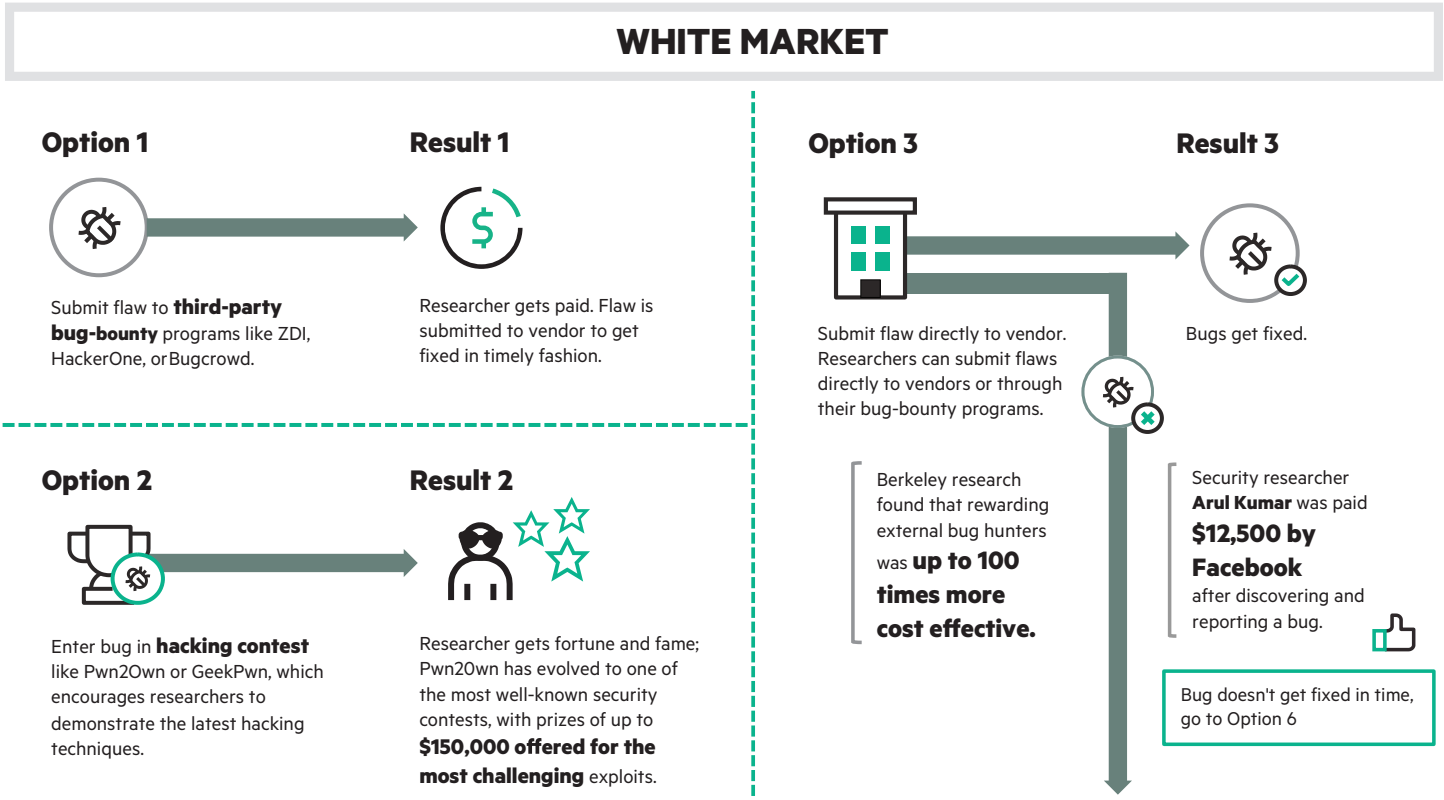


Figure 4. The vulnerability white market

Both vendor programs and the previously mentioned third-party programs operate in the white market. Security researchers submit the vulnerability to either and receive a reward, recognition, or both in trade for their promise to not disclose it—publicly or privately—until the vendor has fixed the flaw. It is understood that the fix should happen in a timely fashion, but ideas differ on just what timely means (Figure 4).

In the gray market, the researcher sells the vulnerability to a private broker. It's often unclear where the flaw will end up and what it will be used for. Some brokers have policies that state they will only sell to ethical and approved sources. Of course, what they consider to be ethical may be different than what others consider to be ethical. Vulnerabilities on the gray market may ultimately be used to spy on private citizens suspected of criminal activities or used to shut down terrorist operations (Figure 5).

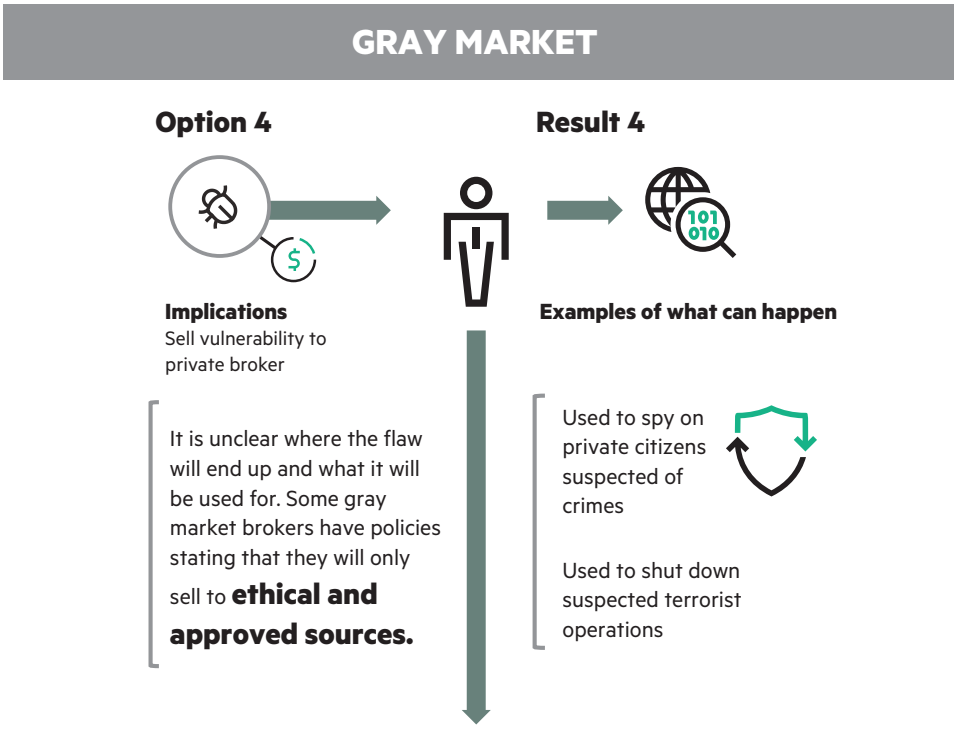


Figure 5. The vulnerability gray market

Historically, the black market has been a method for researchers to sell the vulnerability to the highest bidder with the understanding that it will be used at the sole discretion of the purchaser and likely not for the greater good. Typical outcomes include cybercrime (used to exploit companies in order to steal data or money) and spying (used for political gain or for corporate espionage). More recently, there have been private brokers operating openly on the black market. They pay top dollar for critical wares, such as jailbreaks of the latest version of iOS, and only make these exploits available to their paying customers—never to the software vendor.[13, 14] (Figure 6)

There's always a way to avoid the market altogether by dropping zero days outside of any vendor reporting, an option known as full disclosure (Figure 7). Even in these cases, researchers still monetize the bug by being invited to conferences and increasing their reputational standing in the community.

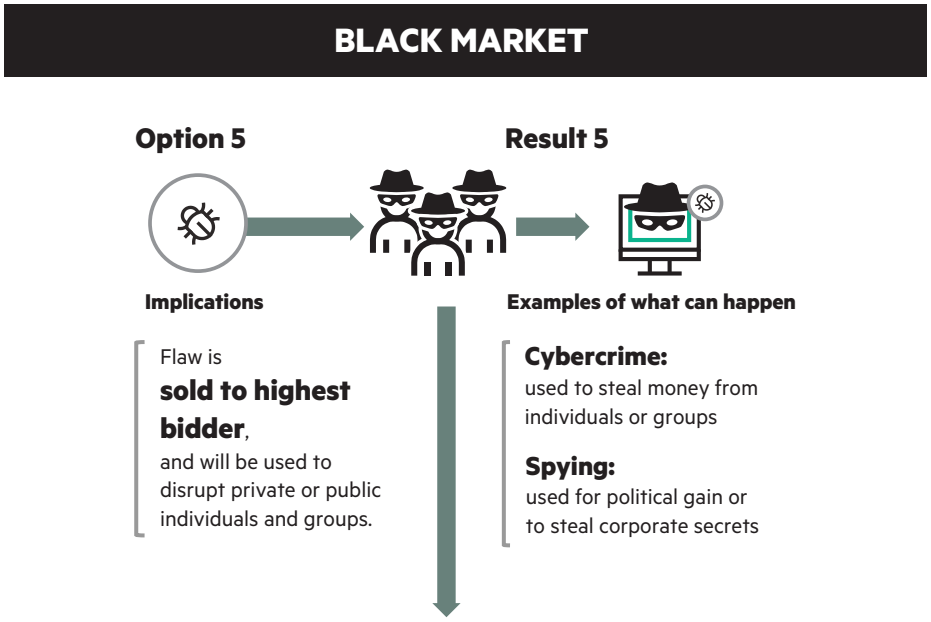


Figure 6. The vulnerability black market

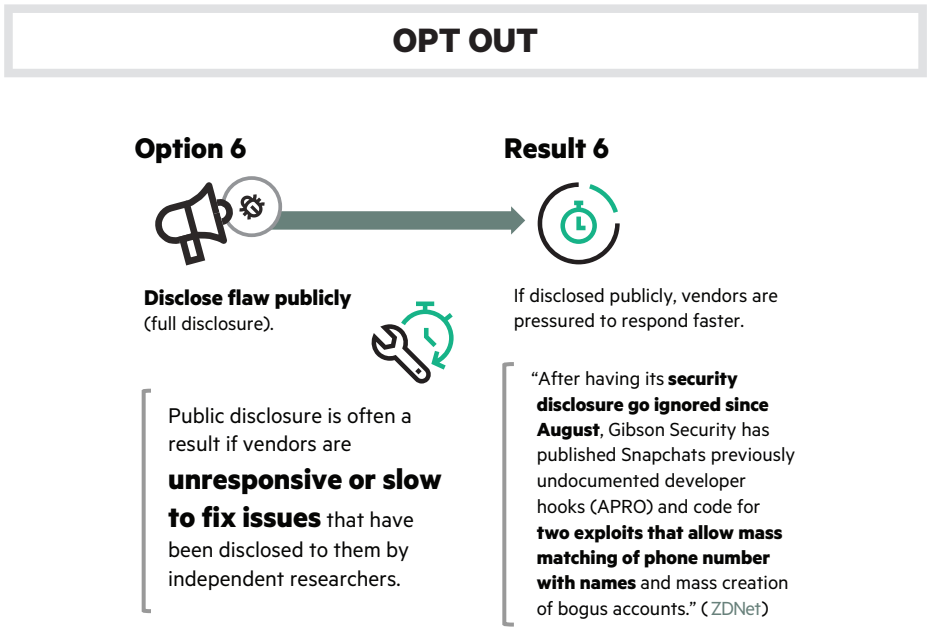


Figure 7. Opting out of the market altogether

[13] http://motherboard.vice.com/read/controversial-zero-day-exploits-seller-launches-new-premium-bug-bounty-program.

[14] http://www.techweekeurope.co.uk/security/zero-day-ios-9-hack-179897.

Pros and cons of market participation

The fundamental elements of trade are buyers and sellers, along with the actual exchange of goods and services. As in any market, if the number of buyers increases, the number of sellers tends to increase as well. In the case where there are incentives for criminal activities, a black or underground market often appears. As long as there is someone willing to pay, there will be someone willing to sell. Security researchers and threat actors seek out vulnerabilities to improve their opportunity for financial gain through the monetization of bugs. What differentiates the two is the market they operate in, as discussed previously. It is assumed that those selling vulnerabilities in the gray and black markets do not execute the exploits themselves out of concern for their own safety. As more and more legislation is being implemented, the risk of prosecution increases. Generally, there are two considerations for selling vulnerabilities on the black market: the financial gain and the risk of being caught by law enforcement. A third possible outcome, in all three markets, is "failure," which occurs if others find and sell/report the same bug. For the researcher—regardless of motivation—it comes down to risk tolerance. All three markets offer increasing rates of return with a correlating increase in risk of running afoul of the law.

Wassenaar effect on security research

Overview

The Wassenaar Arrangement, implemented by more than 40 countries, uses export controls as a means to combat terrorism.[15] The Wassenaar Arrangement means to promote transparency and greater responsibility in the transfer of conventional arms and dual-use technology. The goal in doing so is to prevent destabilizing accumulations of both. Whether or not a transfer is permitted or denied rests with the participating state and not with the governing body. Each participant implements Wassenaar in accordance with its national legislation and policies.[16]

How it already affects research

Where researchers operate in the marketplace is often driven by the country and laws they live under. This is also true for customers in the marketplace. Customers have become wary of the potential consequences of engaging with surveillance companies such as Hacking Team and Gamma International which sell to repressive countries.[17] The recent inclusion of "intrusion software" under the Wassenaar Arrangement seems to be a backlash to offensive security offerings. In May 2015 the US Department of Commerce's Bureau of Industry and Security (BIS) stepped into the fray by offering its proposed implementation of the December 2013 changes for public comment. The proposed implementation of the 2013 changes amended dual-use technologies to include security systems—including intrusion software—for the first time.[18] The BIS proposal included an incredibly broad set of controls related to intrusion software, so broad as to make much of today's defensive cybersecurity research untenable—if not criminal—under the revision. The outcry from the community helped sway BIS into withdrawing its proposed changes and vowing to issue new language in the future.[19]

As an example of the complexities Wassenaar introduces, in 2015 the ZDI worked closely with a number of trade lawyers and government officials to ensure Canada's implementation of the Wassenaar Arrangement was not violated during the annual Pwn2Own contest held at the CanSecWest conference. To do so took many months of security research and communication. With mere weeks to navigate the complexity of obtaining real-time import/export licenses in countries that participate in the Wassenaar Arrangement, the ZDI was unable to sponsor mPwn2Own at PacSecWest in November 2015.

Speculation on the impact to future research

The Wassenaar Arrangement affects the security research community today, and the effects will only increase in the coming years. As the number of cyber-attacks continues to grow, there will likely be a corresponding response by governments to implement laws on how the information security industry operates. Considering the law of unintended consequences—the actions of people/governments always have effects that are unanticipated[20]—we can expect to see increased implementation of the Wassenaar Arrangement and other legislation resulting in decreased efficacy in the security community. The end result means creating a better protection solution becomes harder and takes more time. This, in turn, increases the likelihood of successful breaches as the environment favors those operating in the black market.

[15] http://www.wassenaar.org/introduction/overview.html.

[16] http://www.wassenaar.org/introduction/index.html.

[17] https://tsyrklevich.net/2015/07/22/hacking-team-0day-market/.

[18] https://www.eff.org/deeplinks/2015/05/we-must-fight-proposed-us-wassenaar-implementation.

[19] http://www.privsecblog.com/2015/09/articles/cyber-national-security/pardon-the-intrusion-cybersecurity-worries-scuttle-wassenaar-changes/.

[20] http://www.econlib.org/library/Enc/UnintendedConsequences.html.

> "Infosec has become incredibly important, as recent news amply demonstrates. As a society that depends heavily on technology we need to do much more to ensure that vendors ship securely designed products and are responsive to reports of vulnerabilities." [21]

Moving forward

During the past 20 years, we have watched the world change quite a bit. Just a decade ago, most of the population didn't know what a breach was or that there were careers in cybersecurity. We've seen researchers step into the spotlight and we've seen them shun publicity. There have been laws around research, copyrights, exports, and many other topics. Today, with the "Year of the Breach" just past us, there is more legislation in the US congressional pipeline than ever before, all trying to define "good hackers" and "bad hackers." The vulnerability white market has had a tremendous positive effect in securing the landscape by bringing researchers and vendors together and setting the standard for coordinated disclosure. We expect the white market will continue to evolve as more and more vendors announce their own programs to incentivize research.

We also anticipate regulations and legislation to impact the nature of disclosure. While the environment in which the information security community operates evolves, it is in all of our best interest to continue to find and disclose security bugs in popular software so vendors can fix things in a timely manner. The increasing complexity aside, it continues to be an endeavor we consider worth doing.

[21] http://community.hpe.com/t5/Security-Research/HP-Zero-Day-Initiative-Life-begins-at-10/ba-p/6770464.

The fragility of privacy

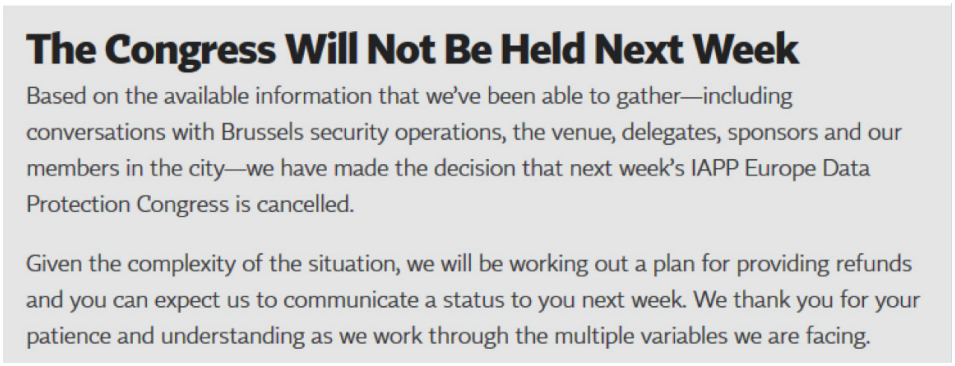
There was perhaps no clearer sign of privacy's fragile situation in 2015 than the notice on the International Association of Privacy Professionals (IAPP) website in November, immediately after the Paris, Kenya, and Beirut bombings.[22]

The IAPP Europe Data Protection Congress 2015, which was scheduled to be held in Brussels during the first week of December, is not an insignificant conference. The topics on its plate this year were mighty: the then-recent upending of the US-EU Safe Harbor agreement; the continuing fallout from Edward Snowden's surveillance revelations in 2013; the role of encryption; and conversations concerning such rising topics as metadata, data localization, the Internet of Things (IoT), data sharing and breach reporting, and more.

And yet it did not happen, and some who had previously cited privacy as their reason for offering certain services used by (among others) the Islamic State/IS terrorists were ceding their ground and changing their services in the face of outrage over their use.[23] By the end of 2015, privacy issues seemed dangerously close to decoupling from security issues in the mind of legislators, the industry, and the public. At what would become a prophetic keynote talk during the Cato Institute's second annual Surveillance Conference in October, Senator Patrick Leahy remarked:

"There are some in Congress who want to give our national security agencies a blank check. They think any attempt to protect our privacy somehow makes us less safe. I hear members accept a framework of 'balancing' privacy rights and national security. But privacy rights are pre-eminent. Protecting our basic privacy rights and protecting our country are not part of a zero-sum equation. We can do both. But we have to keep in mind: If we don't protect Americans' privacy and Constitutional liberties, what have we given up? Frankly I think far too much. And I think this great nation is hurt if we do."[24]

Privacy issues gave the security world much to discuss and ponder throughout 2015.



The Congress Will Not Be Held Next Week

Based on the available information that we've been able to gather—including conversations with Brussels security operations, the venue, delegates, sponsors and our members in the city—we have made the decision that next week's IAPP Europe Data Protection Congress is cancelled.

Given the complexity of the situation, we will be working out a plan for providing refunds and you can expect us to communicate a status to you next week. We thank you for your patience and understanding as we work through the multiple variables we are facing.

Figure 8. Posting on the IAPP website

[22] https://iapp.org/conference/iapp-europe-data-protection-congress-2015/.

[23] https://www.washingtonpost.com/news/morning-mix/wp/2015/11/19/founder-of-app-used-by-isis-once-said-we-shouldnt-feel-guilty-on-wednesday-he-banned-their-accounts/.

[24] http://www.cato.org/events/second-annual-cato-surveillance-conference.

The swamping of Safe Harbor

For enterprises, international data-privacy issues years in the making came to a head in October when Europe's highest court struck down the pact that allowed US and European interests to share data that has privacy considerations, specifically data that includes consumers' personally identifiable information (PII).[25] The EU has safe-harbor relationships with various nation-states; the agreement in effect with the US had been in place since 2000.[26]

The US-EU privacy climate has been tepid since well before Edward Snowden's data releases in June 2013, but the case that tipped the EU justices' scales was a result of Snowden revelations about the Planning Tool for Resource Integration, Synchronization, and Management program, better known as PRISM, a program launched by the National Security Agency (NSA) in 2008.[27] Among the data PRISM gathers is "audio, video and image files, email messages and web searches on major U.S. Internet company websites,"[28] including the likes of Google and Facebook.

Austrian Facebook user Max Schrems filed a complaint stating that Facebook's Irish subsidiary transferred data to the US and thus passed it through PRISM, in contravention of Europe's rigorous privacy protections. The Irish court agreed to look at the matter, and ultimately asked the European Court of Justice whether privacy watchdogs are bound to accept the original declaration that the US is adherent to the standard set for Safe Harbor relationships. The EU court found that the current Safe Harbor arrangement indeed did not adequately protect user privacy rights, because it allowed US officials to gain access to user data even when European law would forbid it and allowed for data to move to third-party nations with which the Safe Harbor agreement was not in force.[29] The Irish data regulator was therefore free to investigate whether the data transfer was properly handled, and Safe Harbor was thus trumped.[30]

Companies on both sides of the Atlantic were in an uproar, scrambling to put together alternate data-transfer mechanisms (that is, mechanisms that are protected by legal devices, such as contractual data-protection clauses, other than the Safe Harbor agreement) even as regulators came knocking.[31] Specialized sectors such as healthcare wondered if they would be able to exchange certain kinds of security research data, while Internet titans such as Google and Facebook were warned[32] by representatives of the Article 29 Working Party, the entity that oversees privacy matters in the EU, not to get "too creative"[33] when plotting end runs around the ruling. Ironically, among the activities planned for the IAPP conference was a lighthearted "S*fe H*rbor Naming Contest" to pre-christen the new arrangement.[34] While this contest drew some creative entries,[35] it was later announced that the group was recommending that the new arrangement be called "The Transatlantic Data Protection Framework."[36]

The US Department of Commerce, with which the original agreement was negotiated and which had been working for two years prior to nail down a stronger agreement, termed itself "deeply disappointed" and vowed to work for a rapid upgrade to the problematic frameworks.[37] The US House of Representatives passed the Judicial Redress Act giving certain foreign citizens the right to sue over US privacy violations related to shared law enforcement data, which chief sponsor Jim Sensenbrenner said explicitly could help to mend US-EU fences.[38]

As this Risk Report went to press, the target date for a new framework was January 31, 2016. If no solution is found by then, "EU data protection authorities are committed to take all necessary and appropriate actions, which may include coordinated enforcement actions."[39] According to at least one European official, the likelihood of a solution in that time frame was not good,[40] in which case business slowdowns and even very large fines would ensue.

[25] https://www.washingtonpost.com/world/national-security/eu-court-strikes-down-safe-harbor-data-transfer-deal-over-privacy-concerns/2015/10/06/2da2d9f6-6c2a-11e5-b31c-d80d62b53e28_story.html.

[26] Op. cit.

[27] http://uspolitics.about.com/od/antiterrorism/a/What-Is-Prism-In-The-National-Security-Agency.htm.

[28] Op. cit.

[29] http://www.law360.com/privacy/articles/711346.

[30] http://www.law360.com/privacy/articles/716286.

[31] http://www.law360.com/privacy/articles/711385.

[32] http://www.dataprotectionreport.com/2015/10/wp29-issues-post-safe-harbor-guidance/.

[33] http://www.law360.com/privacy/articles/716493/eu-watchdog-says-creativity-not-answer-to-data-pact-demise.

[34] https://iapp.org/news/a/and-the-winner-is/.

[35] https://iapp.org/news/a/sfe-hrbor-naming-contest-the-final-round.

[36] https://iapp.org/news/a/and-the-winner-is/.

[37] https://www.commerce.gov/news/press-releases/2015/10/statement-us-secretary-commerce-penny-pritzker-european-court-justice.

[38] http://judiciary.house.gov/index.cfm/2015/10/goodlatte-sensenbrenner-and-conyers-praise-house-passage-of-legislation-to-strengthen-privacy-protections-for-individuals.

[39] https://www.technologyslegaledge.com/2015/10/breaking-news-safe-harbor-g29-issues-its-first-statement-on-schrems/.

[40] http://www.theregister.co.uk/2015/12/01/safe_harbor_solution_not_soon/.

Surveillance

Ironically, the beginning of 2015 promised positive privacy developments, as observers awaited the sunsetting of NSA bulk data collection authority originally granted by 2001's Patriot Act.[41] The powers granted by Congress in the wake of 9/11 were vast. In the years after 9/11, the tide of judicial and public opinion had turned against what many saw as vast overreach and even vaster failure to perform. In May, the US Second Circuit Court ruled that the program to systemically collect Americans' phone records—specifically, the clause known as Section 215[42]—had never been properly authorized.[43] The Patriot Act expired on June 1, and Section 215 with it. On June 2, Congress approved the USA Freedom Act, which included a ban on those collection activities.[44] Various Congressional attempts[45] to restore the program were unsuccessful, and the ban took effect on November 29.[46]

Even as the NSA has struggled to give the public and Congress more transparency into its workings,[47] evaluations indicate that the bulk-collection program was a failure. A number of investigations[48] by various government committees[49] and other observers[50] described a broken program with no provable success at pinpointing data applicable to the stated task—that is, protecting Americans from attack. The most successful case spotted in the data—that of a Somali man convicted of sending $8500 to a group in his home country—involved no threat of attack against the US.[51]

Even the agencies themselves seemed nonplussed by the results of bulk data collection. At a Cato Institute event, senior fellow John Mueller speaking on the efficacy of the programs ("Surveilling Terrorists: Assessing the Costs and Benefits") noted that the data has been proven remarkably ineffectual at spotting terrorists.[52] Moreover, he said, the very facts of the tidal wave of data, coupled with the "9/11 Commission Syndrome" expectation that every lead must be followed up regardless of implausibility, has led to high levels of conflict between agencies, which resent the low-quality and irrelevant leads derived from the data.[53] He noted a troubling brain-drain cost as good investigators become increasingly hopeless and paranoid as a byproduct of pointless "protection" efforts.[54]

> "Protecting our privacy rights and protecting our country are not part of a zero-sum equation. We can do both."
>
> - Sen. Patrick Leahy (D-VT)

[41] http://thehill.com/blogs/pundits-blog/homeland-security/243169-a-beautiful-sunset-provision-for-nsa-surveillance.

[42] http://www.nytimes.com/2015/05/08/us/nsa-phone-records-collection-ruled-illegal-by-appeals-court.html.

[43] http://pdfserver.amlaw.com/nlj/NSA_ca2_20150507.pdf.

[44] http://judiciary.house.gov/index.cfm/usa-freedom-act.

[45] http://www.theguardian.com/us-news/2015/jun/03/nsa-surveillance-fisa-court.

[46] http://www.npr.org/sections/thetwo-way/2015/11/29/457779757/nsa-ends-sept-11th-era-surveillance-program.

[47] https://www.nsa.gov/public_info/declass/IntelligenceOversightBoard.shtml.

[48] https://www.propublica.org/article/whats-the-evidence-mass-surveillance-works-not-much.

[49] https://www.washingtonpost.com/world/national-security/nsa-shouldnt-keep-phone-database-review-board-recommends/2013/12/18/f44fe7c0-67fd-11e3-a0b9-249bbb34602c_story.html.

[50] https://www.newamerica.org/international-security/do-nsas-bulk-surveillance-programs-stop-terrorists/.

[51] https://www.washingtonpost.com/world/national-security/nsa-phone-record-collection-does-little-to-prevent-terrorist-attacks-group-says/2014/01/12/8aa860aa-77dd-11e3-8963-b4b654bcc9b2_story.html.

[52] http://www.cato.org/events/second-annual-cato-surveillance-conference.

[53] Op. cit.

[54] Op. cit.

But the attacks in Paris and the proximity of an American election year were powerful enough to bring bulk-collection surveillance proposals back to "life." While Paris struggled back to normalcy, one Republican (GOP) candidate was already backing a call by an Arkansas senator to reinstate bulk collection through January 2017.[55] The proposal drew heavy fire from others on the GOP slate, with Rand Paul choosing particularly strong language to express his opposition.[56] Elsewhere in Congress, one Democratic senator attempted[57] to add provisions to the high-profile Cybersecurity Information Sharing Act (CISA) bill that would add unvetted new "capabilities" for law enforcement seeking data access.[58] A long-running Federal Bureau of Investigation (FBI) program utilizing National Security Letters that allows for mass warrantless seizure of data was revealed late in the year.[59] Most concerning is a report by The New York Times that the NSA has, after all, found a way around the sunsetting—and the limited judicial oversight provided in the Patriot Act—and is gathering all the data it wants simply by mass foreign collection.[60]

It should be understood that surveillance is not only on the rise in America. The next section details various international developments, but it would be unfair to leave our surveillance section without mentioning recent work by entities looking to monitor and engage on the issues worldwide. Notably, July saw the release of AccessNow's excellent "Universal Implementation Guide for the International Principles on the Application of Human Rights to Communications Surveillance,"[61] which provides implementation guidance for the information set forth in 2014 on the Necessary and Proportionate site.[62] That site, an Electronic Frontier Foundation project, documents ongoing efforts to reconcile existing human rights law to modern surveillance technologies. It's all very relevant as governments worldwide trend toward greater surveillance of citizens.

Encryption

If surveillance manages time and again to seem like a white knight after terrorist incidents, encryption is often the dragon. In the days after the Paris attacks, various simmering encryption-related debates were back on the boil, despite early evidence (still under investigation) that encryption played no role in the terrorists' planning.[63]

The United Kingdom was already dealing with rushed[64] calls by legislators for Internet providers and social-media sites to provide unencrypted access and/or backdoors to encrypted communications to law enforcement and spy agencies.[65] By the end of the year some American legislators were making similar calls,[66] stating that law enforcement is unable to access necessary data. Those arguments were countered by equally venerable arguments by crypto experts[67] about the certainty that backdoors—or, worse, giant stores of unencrypted data—are a recipe for unwanted, sustained, and ultimately catastrophic attention from attackers.[68] At the time of this Report's writing, Senator Ron Wyden (D-OR) was brushing off his proposed 2014 Secure Data Act,[69] which seeks to ban government-mandated tech backdoors.[70] One hardware manufacturer left an entire market rather than bend to government demands for unfettered backdoor access, as BlackBerry prepared to leave the Pakistan market at year's end rather than expose its BlackBerry Enterprise Service (BES) traffic to wholesale traffic monitoring.[71]

[55] http://www.vnews.com/news/nation/world/19713942-95/arkansas-senator-trying-to-extend-bulk-phone-data-collection.

[56] http://blogs.rollcall.com/wgdb/rand-paul-surveillance-rubio-cruz-cotton/.

[57] http://www.law360.com/privacy/articles/716526/groups-slam-bid-to-use-cybersecurity-bill-to-expand-cfaa.

[58] http://www.law360.com/privacy/articles/715474.

[59] http://www.zdnet.com/article/fbi-can-force-companies-to-turn-over-user-data-without-a-warrant/.

[60] http://www.nytimes.com/2015/11/20/us/politics/records-show-email-analysis-continued-after-nsa-program-ended.html.

[61] https://s3.amazonaws.com/access.3cdn.net/a8c194225f95db00e9_blm6ibrri.pdf.

[62] https://necessaryandproportionate.org/.

[63] https://www.techdirt.com/articles/20151118/08474732854/after-endless-demonization-encryption-police-find-paris-attackers-coordinated-via-unencrypted-sms.shtml.

[64] http://www.theregister.co.uk/2015/11/26/mps_and_peers_have_just_weeks_to_eyeball_uk_govs_supersnoop_bid/.

[65] http://www.telegraph.co.uk/news/uknews/terrorism-in-the-uk/11970391/Internet-firms-to-be-banned-from-offering-out-of-reach-communications-under-new-laws.html.

[66] http://techcrunch.com/2015/11/24/the-encryption-debate-isnt-taking-a-thanksgiving-break/.

[67] http://passcode.csmonitor.com/influencers-paris.

[68] http://www.thedailybeast.com/articles/2015/11/30/feds-want-backdoor-into-phones-while-terrorists-walk-through-front-door.html.

[69] https://www.wyden.senate.gov/news/press-releases/wyden-introduces-bill-to-ban-government-mandated-backdoors-into-americans-cellphones-and-computers.

[70] https://medium.com/backchannel/encryption-is-not-the-enemy-b5c1652e30b8#.vmnu2lnj1.

[71] http://mashable.com/2015/11/30/blackberry-pakistan-exit/.

Information sharing

There are positive ways of sharing threat information, of course, and some progress was made to build those systems. In February, President Obama signed Executive Order 13691,[72] which details a framework to expand both private sector and public-private sharing of information on threats and attacks. Such programs have long been the goal of a number of public and private efforts. The STIX and TAXII framework standards, for instance, have been around for years,[73] while several commercial entities have attempted to build the infrastructure and attract the critical mass necessary to make such entities a going concern. Over the course of the year, the House and Senate worked on legislation[74] that would protect companies engaged in such sharing, though as mentioned above at least one senator made an attempt to piggyback a surveillance project onto the Senate offering.[75] By the end of the year, the Information Sharing and Analysis Organization (ISAO) standards group was holding public meetings to discuss next steps,[76] and various state[77] and local[78] entities were examining how they might participate, whether by standing up their own ISAO groups as suggested by the Executive Order or via some other means.

Spotlight: three tech giants

Many of the issues discussed so far in this section center on broad, nation-state-type entities, but these issues also tended to touch the largest technology businesses. The 2015 corporate accountability index published by Ranking Digital Rights found that none of the world's highest-profile tech firms were particularly trusted by their users. According to the results of the survey, the customers found them lacking in transparency, inconsistent in their privacy disclosures, and generally ranging from not-great to just awful.[79]

In previous years, such companies as Microsoft, Google, and Facebook had various incidents in privacy, and 2015 brought new variations on that theme. We will touch on how other sectors were affected (mainly by regulation and legislation) in the next section, but for now let's take a look at some of the issues these three corporate giants faced in 2015.

[72] http://www.law360.com/articles/621688.

[73] http://stixproject.tumblr.com/post/119254803262/a-history-of-stixtaxiicyboxmaec-news-media.

[74] http://www.law360.com/publicpolicy/articles/728705.

[75] http://www.law360.com/privacy/articles/716526/groups-slam-bid-to-use-cybersecurity-bill-to-expand-cfaa.

[76] https://fcw.com/articles/2015/11/09/information-sharing-isao.aspx.

[77] http://www.law360.com/privacy/articles/645293.

[78] http://www.law360.com/articles/699517/dhs-grants-ut-san-antonio-11m-for-info-sharing-standards?article_related_content=1.

[79] http://www.theguardian.com/technology/2015/nov/03/data-protection-failure-google-facebook-ranking-digital-rights.

Spotlight: Google

Google spent much of its year addressing requests to remove well over one million URLs from search results in the wake of the EU's May 2014 "right to be forgotten" ruling (Google Spain v AEPD and Mario Costeja Gonzalez).[80] At the time of this writing, the company had received over 350,000 requests and evaluated over 1.2 million URLs for removal.[81] The company has declined just over half of those requests[82] and publicly documented its progress.[83] Interestingly, the domain most frequently cited in removal requests appears to be Facebook. Google has also been fighting privacy-violations claims in a suit consolidating two dozen smaller, similar suits. Plaintiffs in the case claim that the site surreptitiously bypasses user privacy settings and collects information to which it should not be a party. The case has been dismissed by a lower court and remained dismissed on appeal to the Third Circuit.[84]

Spotlight: Microsoft

Microsoft started the year already embroiled in a case involving customer emails stored on a server in Ireland. In July 2014, the company was requested to turn over data on a customer whose data resided on its Irish subsidiary's servers. Microsoft resisted, saying that the warrant did not apply to electronic communications stored outside the US. The company has been in court ever since and presented its argument to the Second Circuit in September.[85] The outcome, especially with Safe Harbor now in play, remains to be seen.[86] In that suit, the company is arguing a position that includes protection of its customers' data.

Spotlight: Facebook

Facebook is experiencing interesting challenges with regard to its use of information this year. Aside from Safe Harbor, the company is respondent in a class-action case in California in which users claim the service scans information in private messages for profit.[87]

Facebook's "Real Name" policy of requiring users to provide and display their legally registered names appears to be controversial. In October, more than six dozen activist groups penned an open letter asking the service to rethink its policy.[88] The company says it is in the process of reworking the policy,[89] but will retain it in some form.[90]

Germany contemplated legal action over anti-migrant hate speech Facebook allowed to remain on its Walls.[91] Lawsuits are arising around Facebook's facial recognition system, which may violate various jurisdictions' rules about storing biometric data without permission.[92] In this situation, Facebook is not alone. Illinois state law has been actively cited in other possible violations of this law, most notably by Shutterfly[93] and by videogame house Take-Two Interactive Software.[94] In the latter case, the company is accused of capturing and storing 3D scans of players' faces and of making them visible to other users online without permission—surely a new dimension to the problem of biometric privacy.

[80] http://curia.europa.eu/juris/document/document.jsf?docid=152065&doclang=en.

[81] http://www.google.com/transparencyreport/removals/europeprivacy/.

[82] http://www.law360.com/privacy/articles/731696.

[83] http://www.google.com/transparencyreport/removals/europeprivacy/.

[84] http://www.law360.com/articles/737289/3rd-circ-denies-rehearing-bid-in-google-tracking-suit.

[85] http://www.law360.com/privacy/articles/699043.

[86] https://www.washingtonpost.com/news/volokh-conspiracy/wp/2015/07/23/does-it-matter-who-wins-the-microsoft-ireland-warrant-case/.

[87] http://www.law360.com/privacy/articles/723008.

[88] http://www.independent.co.uk/life-style/gadgets-and-tech/news/facebook-to-tweak-real-name-policy-after-backlash-from-lgbt-groups-and-native-americans-a6717061.html.

[89] http://spectrum.suntimes.com/news/10/155/5716/facebook-real-name-policy-lgbt-community.

[90] https://nakedsecurity.sophos.com/2015/11/03/facebook-finally-changes-real-name-policy/.

[91] http://money.cnn.com/2015/11/11/news/companies/facebook-germany-hate-posts/

[92] http://www.natlawreview.com/article/facebook-seeks-dismissal-illinois-facial-recognition-biometric-privacy-suit.

[93] http://www.law360.com/privacy/articles/703969.

[94] http://www.law360.com/privacy/articles/715863.

Legislation and regulation

The Federal Trade Commission (FTC), which in 2014 made a strong play to lead on federal cyber policy issues, had a busy year. The high-profile FTC v Wyndham Worldwide Corporation case[95] that we discussed in this space last year continued to rack up wins for the Commission as the Third Circuit affirmed[96] that the agency has the authority to regulate cybersecurity as a function of the "unfairness prong" of section 45 of The FTC Act.[97, 98] In mid-December, Wyndham agreed to settle the charges and establish an information security program compliant with the Payment Card Industry Data Security Standard (PCI-DSS) and to accept audits of the program for the next 20 years. If the court accepts the settlement proposal, the case will be a major signal to businesses that the FTC is the agency to watch when pondering enterprise cyber-obligations.[99] On the other hand, the agency's case against LabMD for insufficient data protection was dismissed by a judge for the US District Court for Washington, D.C., in 2013.[100] The CEO for the now-defunct company immediately brought suit against three FTC lawyers accusing them of, among other things, building their case on "lies, thievery and testimony" supplied by a third party,[101] which LabMD is also suing.[102] The FTC at this writing had requested internal review of the ruling.[103]

The agency has been forging a stronger relationship with the Federal Communications Commission's (FCC) own security and privacy teams,[104] which, as one FTC commissioner phrased it, have been a "brawnier cop on the privacy beat" to this point.[105] She also noted the FTC is still leading the squad,[106] though not without internecine conflict. In other words, don't rule out that turf war just yet.[107] The FTC also welcomed as its new chief technologist Jonathan Meyer, highly regarded in privacy circles for his research on online tracking by such companies as Google and Verizon, as well as for his development of various anti-tracking browser mechanisms.[108] Other strong FTC privacy concerns included consumer tracking by marketers (particularly when the tracking isn't opt-in), trust in advertising, unwanted data collection,[109] and liability for companies that farm out their data security.[110]

[95] https://www.ftc.gov/news-events/blogs/business-blog/2015/08/third-circuit-rules-ftc-v-wyndham-case.

[96] http://www.natlawreview.com/article/third-circuit-sides-ftc-data-security-dispute-wyndham.

[97] http://www.gpo.gov/fdsys/pkg/USCODE-2011-title15/pdf/USCODE-2011-title15-chap2-subchapI-sec45.pdf.

[98] http://www.hldataprotection.com/2015/08/articles/consumer-privacy/analysis-of-ftc-v-wyndham-third-circuit-affirms-ftc-authority-to-regulate-data-security/.

[99] http://www.reuters.com/article/us-wyndham-ftc-cybersecurity-idUSKBN0TS24220151209#PRPyEFzQXsCj1q4P.97.

[100] http://www.healthdatamanagement.com/news/FTC-data-security-complaint-against-LabMD-dismissed-51615-1.html.

[101] http://www.law360.com/articles/731134/labmd-sues-3-ftc-lawyers-over-data-security-case.

[102] http://www.law360.com/privacy/articles/733023.

[103] http://www.law360.com/privacy/articles/731601.

[104] http://www.law360.com/articles/727540/fcc-teaming-up-with-ftc-on-consumer-protection.

[105] http://www.law360.com/publicpolicy/articles/729885.

[106] http://www.law360.com/articles/708010/ftc-still-top-privacy-cop-despite-fcc-order-brill-says.

[107] http://www.law360.com/privacy/articles/734946.

[108] https://jonathanmayer.org/.

[109] http://www.law360.com/privacy/articles/708001.

[110] http://www.natlawreview.com/article/piercing-outsourcing-veil-ftc-says-data-security-obligations-remain.

It was, as every year for years has been, a year of new records.

Some US federal agencies resisted calls to promptly release guidance,[111, 112] while others stepped up in their various spheres. The Department of Defense (DoD) in September released new cybersecurity regulations in the wake of the OPM breach. These new regulations covered everything from breach reporting to log retention to cloud-related issues.[113] Later in the fall, it updated rules pertaining to how IT contractors are brought into their supply chain and how sensitive information may be shared with them.[114] The Department of Commerce (DoC) continued to fine-tune proposed rules controlling the export of hacking tools; an open-comment period in May drew a large response[115] from security researchers and firms already feeling a Wassenaar[116]-related chill in the air. Commerce has so far given no date for release of a revised ruleset.[117] Meanwhile, the Department of Homeland Security (DHS) put out an end-of-year call for applicants to three-year appointments to its Data Privacy and Integrity Advisory committee.[118] By year's end, the federal Office of Management and Budget announced a Federal Privacy Council that will coordinate privacy policies and strategies across multiple government agencies.[119]

Beyond the US, international nation-specific efforts to think about data privacy (and surveillance, and encryption) continued even as the US-EU Safe Harbor situation unspooled, and a full recounting on them is beyond the scope of this Risk Report. Cross-border efforts to reach rule consensus[120] increased as legislators and the general public worldwide finally got a look at the text of the Trans-Pacific Partnership,[121] which will be a major factor in 2016's privacy story. However, the trend toward data localization—that is, to require citizens' data to reside in territory controlled by the nation of which they are citizens—will likely affect privacy-protection efforts in new ways.[122] Earlier this year, Australia accused China of hacking its Bureau of Meteorology. Over the years Australia's media offices, power grids, and intelligence headquarters have allegedly come under attack from the same quarter.[123] And to complete our circumnavigation of the globe, in September the US and China signed a bilateral anti-cyber espionage accord that raised eyebrows inside government and beyond.[124, 125]

Breaches in the news

If 2014 was the Year of the Breach, 2015 was the Year of Collateral Damage, as certain attacks touched people who never dreamed they might be present in, or identifiable from, the data involved.

It was, as every year for years has been, a year of new records. The January Anthem breach drew headlines for affecting 80 million records.[126] By November, a banking breach affecting 100 million accounts passed nearly without a trace in the headlines.[127] Anthem was reduced to guest appearances in other healthcare-related breach coverage[128] and in background material on the OPM breach, which has been attributed to the same attackers.[129] A recounting by someone affected by last year's Sony breach, ironically, seemed to make the rounds far more widely.[130] By year's end, a weary observer could see a headline about a potential breach of six million voter records in Georgia and merely think it was odd that the reporter described it as "massive."[131] It seemed as if everyone was getting hit, and repeatedly. Victims ranged from the unsympathetic[132] to the criminal[133] to the complex but well-trod ethical middle ground of "folks who ought to know better." [134, 135, 136]

[111] http://www.commlawblog.com/2014/04/articles/broadcast/drone-even-go-there-on-newsgathering-drones-and-the-faa/.

[112] http://www.chicagotribune.com/news/opinion/commentary/ct-drones-privacy-laws-20150803-story.html.

[113] http://www.law360.com/privacy/articles/705295.

[114] http://www.law360.com/privacy/articles/721375/pentagon-finalizes-cyber-risk-rule-for-sensitive-it-contracts.

[115] http://www.theregister.co.uk/2015/07/30/us_to_rethink_wassenaar/.

[116] http://www.wassenaar.org/publicdocuments/index.html.

[117] http://www.law360.com/privacy/articles/702478.

[118] https://s3.amazonaws.com/public-inspection.federalregister.gov/2015-30545.pdf.

[119] https://iapp.org/news/a/federal-government-announces-federal-privacy-council/.

[120] http://www.law360.com/privacy/articles/699125.

[121] https://ustr.gov/tpp/.

[122] http://www.law360.com/privacy/articles/698895.

[123] http://shanghaiist.com/2015/12/03/china_hacks_australian_weather_bureau.php.

[124] http://atimes.com/2015/11/counterintelligence-chief-skeptical-that-china-has-curbed-spying-on-us/.

[125] http://arstechnica.com/tech-policy/2015/09/analysis-china-us-hacking-accord-is-tall-on-rhetoric-short-on-substance/.

[126] http://www.therecorder.com/id=1202743330885/Anthem-Fires-Back-at-Data-Breach-Suit?slreturn=20151030065146.

[127] http://mashable.com/2015/11/10/bank-data-breach-100-million/#5TwzKTwTuOqh.

[128] http://www.democratandchronicle.com/story/news/2015/09/09/excellus-announces-august-breach-system/71949658/.

[129] http://www.reuters.com/article/2015/12/02/us-china-usa-cybersecurity-idUSKBN0TL0F120151202#dZk1fXJZmkx22AXh.97.

[130] http://www.slate.com/articles/technology/users/2015/11/sony_employees_on_the_hack_one_year_later.single.html.

[131] http://www.ajc.com/news/news/state-regional-govt-politics/suit-accuses-georgia-of-massive-data-breach-involv/npQLz/.

[132] http://pastebin.com/Yh2muT9r.

[133] http://www.law360.com/privacy/articles/703036.

[134] https://www.htbridge.com/advisory/HTB23282.

[135] http://www.theregister.co.uk/2015/11/25/dsdtestprovider.

[136] http://www.theguardian.com/technology/2015/oct/19/cia-director-john-brennan-email-hack-high-school-students.

Many consumers are inured these days to breach notifications from credit-card companies and the odd medical clinic, but 2015 brought us attackers who tried to extort crowdfunded artists[137] and, more benignly, found ways to turn our teakettles[138] and "smart" homes[139, 140] against us. And yet these breaches in turn paled when attackers breached V-Tech's customer database,[141] which included images of customers and their children.[142] Predictably,[143] others hijacked a Wi-Fi-enabled incarnation of Barbie.[144]

Even these breaches were perhaps not the most chilling of the year, even if they did target children and musicians and other relatively harmless folk—because even with all that, the kid possesses the toy, the musician benefits from the crowdfunding account, the homeowner owns the thermostat. There are, however, two 2015 breaches that best demonstrate that personal privacy violations can be perfectly impersonal: the OPM breach and the notorious Ashley Madison hack and data blast.

The OPM breach, which hijacked data of over 21 million current and former federal employees, took place in mid-2014 and was revealed last spring.[145] Reports indicate that a specific nation-state is believed to have stolen that data,[146] though that nation-state denies[147] the breach was state-sponsored. The bulk of the action took place in a quiet, intense cat-and-mouse game, with the affected parties learning details after the fact. In contrast, the Ashley Madison breach[148] was deliberately loud and messy—a previously unknown hacker, claiming moral authority over both the site's customers and its business operations,[149] unleashed a tidal wave of intensely personal data[150]—in addition to the startling-to-most fact that an adultery-matchmaking site had 32 million registered accounts, though perhaps not all of them operated by actual humans.[151]

These breaches don't initially look the same; however, both breaches had terrible effects on people who never had direct contact with the keepers of the data, and whose information appeared in it only as it related to someone else—or, in the case of the Ashley Madison breach, did not appear at all but whose identity could be easily deduced from revealed data (e.g., a spouse's name and address would be knowable to a nosy neighbor if one spouse was registered on the site under his or her true name[152]).

[137] http://techcrunch.com/2015/11/21/extortionists-are-threatening-to-release-patreon-user-data/.

[138] http://boingboing.net/2015/10/23/putting-your-kettle-on-the-int.html.

[139] https://securityledger.com/2015/11/green-light-or-no-nest-cam-never-stops-watching/.

[140] https://www.abiresearch.com/press/nest-cam-works-around-clock/.

[141] http://www.theverge.com/2015/11/27/9807330/vtek-data-breach-password-email-address.

[142] http://arstechnica.com/security/2015/11/hacked-toymaker-leaked-gigabytes-worth-of-kids-headshots-and-chat-logs/.

[143] http://www.theguardian.com/technology/2015/mar/13/smart-barbie-that-can-listen-to-your-kids-privacy-fears-mattel.

[144] http://www.theguardian.com/technology/2015/nov/26/hackers-can-hijack-wi-fi-hello-barbie-to-spy-on-your-children.

[145] https://fcw.com/articles/2015/08/21/opm-breach-timeline.aspx.

[146] http://www.nytimes.com/2015/08/01/world/asia/us-decides-to-retaliate-against-chinas-hacking.html.

[147] http://www.reuters.com/article/2015/12/02/us-china-usa-cybersecurity-idUSKBN0TL0F120151202.

[148] http://fortune.com/2015/08/26/ashley-madison-hack/.

[149] Op. cit.

[150] http://www.theverge.com/2015/8/19/9179037/ashley-madison-data-hack-name-address-phone-birthday.

[151] http://techcrunch.com/2015/08/31/ashley-madison-refutes-claims-that-its-site-was-populated-with-fake-female-accounts/.

[152] http://googlemapsmania.blogspot.com/2015/08/ashley-madison-users-mapped.html.

The Ashley Madison situation revealed new levels of negative effect as individuals reacted to having the information put on blast. Stories of firings,[153] grief,[154] divorce,[155] ruin,[156] and suicide[157] were all over the news, thereby intensifying public scrutiny of many people who had no business relationship to Ashley Madison. One observer noted that certain effects could be even more cataclysmic, because active-duty members of the military found in the database could be subject to dishonorable discharge—meaning that unemployment and loss of pension are genuine possibilities.[158]

Despite the three years of credit counseling offered to persons whose names were revealed in the OPM hack, it's a relatively good bet that the stolen data wasn't meant for the hands of criminal gangs or identity thieves. Instead, the OPM hack bore a resemblance to a rash of hacks against newspaper reporters a couple of years ago[159]—hacks that sought the names of reporters' contacts, most likely those contacts who are dissident to a particular government.[160] It is believed that within the rich trove of data taken were tens of thousands of Standard Form (SF)-86s, which are filled out by any service member or civilian who seeks a security clearance.[161]

As those who have gone through that screening are aware, one provides a great deal of information on the SF-86 about one's family, friends, and associates[162]—for security and intelligence professionals, a delicate situation. In other words, the true targets of the breach may, again, be people who never themselves consented to inclusion in the OPM database—and who may be in danger thanks to its compromise. (It is estimated that the potential for damage could last for over 40 years.[163]) In at least one case, it was decided that a number of CIA officials covertly stationed in a particular embassy needed to be pulled precisely because they did not appear in OPM files, as genuine state employees would.[164]

Déjà vu again

A handful of 2015 incidents seemed to have returned from a previous calendar. Remember when Radio Shack insisted on gathering too much personal information at the register?[165] This year it was a clothing retailer doing it instead.[166] Remember 2004, when the popular karaoke jam was Outkast's "Hey Ya!"[167] and Calyx Internet Access received a National Security Letter it decided to fight in the courts? That's only just been settled.[168] In the meantime, a federal court ordered the release of the information requested by an actual NSL.[169] How about 2008, when the economy cratered and a recruiter ended up in court for "hacking" a database to which he had a legitimately acquired a password? Still in the courts.[170] Remember when we used to worry that we were being tracked for marketing purposes by the mobile phones in our pockets? We were.[171]

Following the notorious Target breach, the company was back this holiday season with a $39 million settlement to be paid to all the financial institutions that had to scramble on sending new cards to their customers. This also marks the first successful class-action suit by financial institutions against a breached company.[172] A Massachusetts jeweler risked the phenomenon known as the Streisand Effect[173] when it took Yelp to court to demand the service reveal the name of a user who submitted a particularly angry review.[174] This keeps happening[175] and petitioners will keep looking for a venue that will throw out the portion of the Communications Decency Act that lets sites off the hook for allegedly libelous statements by users.[176] The past isn't dead; it isn't even the past.[177]

[153] http://nypost.com/2015/12/06/ashley-madison-hack-steals-mans-job-wife-and-mind/.

[154] http://www.huffingtonpost.com/drs-bill-and-ginger-bercaw/the-sad-irony-of-the-ashl_b_7868828.html.

[155] http://fusion.net/story/185647/ashley-madison-hack-victims/.

[156] http://money.cnn.com/2015/08/21/technology/ashley-madison-ruined-lives/.

[157] http://hollywoodlife.com/2015/09/09/ashley-madison-suicide-married-baptist-pastor-john-gibson/.

[158] Private conversation with retired military person who requests anonymity.

[159] http://www.nytimes.com/2013/02/02/technology/washington-posts-joins-list-of-media-hacked-by-the-chinese.html.

[160] http://www.theguardian.com/media/2013/jan/31/new-york-times-chinese-hacked.

[161] http://www.navytimes.com/story/military/2015/06/17/sf-86-security-clearance-breach-troops-affected-opm/28866125/.

[162] https://www.opm.gov/forms/pdf_fill/sf86.pdf.

[163] http://fedscoop.com/opm-losses-a-40-year-problem-for-intelligence-community.

[164] http://arstechnica.com/tech-policy/2015/09/cia-officers-pulled-from-china-because-of-opm-breach/.

[165] http://www.pcworld.com/article/2925352/radioshack-us-states-reach-agreement-on-sale-of-customer-data.html.

[166] http://www.law360.com/privacy/articles/634393.

[167] http://www.bobborst.com/popculture/top-100-songs-of-the-year/?year=2004.

[168] http://www.law360.com/privacy/articles/703542.

[169] http://arstechnica.com/tech-policy/2015/11/the-national-security-letter-spy-tool-has-been-uncloaked-and-its-bad./

[170] http://www.law360.com/privacy/articles/715447/ex-kfi-recruiter-takes-cfaa-charges-to-9th-circ-again.

[171] https://www.ftc.gov/news-events/press-releases/2015/09/ftc-approves-final-order-nomi-technologies-case.

[172] http://www.law360.com/privacy/articles/733321.

[173] https://en.wikipedia.org/wiki/Streisand_effect.

[174] http://www.law360.com/privacy/articles/702982.

[175] http://www.law360.com/media/articles/732112.

[176] http://www.law360.com/media/articles/732112.

[177] https://en.wikipedia.org/wiki/Requiem_for_a_Nun.

A look ahead

We will be contending with the events of 2015 for some time and 2016 will bring its own excitements. In addition to the Safe Harbor revamp, expect to see activity around the meaning and uses of metadata, the development of the Internet of Things, continued controversy in the worlds of encryption and security, fresh efforts to contain certain kinds of online abuses, and maybe progress in bringing what we've all learned about data privacy to bear in the wider world.

Expect to hear from people looking for a more nuanced understanding of metadata and how much it reveals. At Columbia University, a team of researchers led by Steven Bellovin and Stephanie Pell has been examining whether our current concept of metadata takes into proper account how much actual information can be derived from the means and paths of communications, even when the observer is not privy to the specific contents of the communication. For example, if someone were to look at Alice's Internet history and see that she visited one of the Ashley Madison breach data-search sites, followed by web pages such as divorce-that-loser.org, followed three months later by Tinder and Zillow.com, a good guess could be made as to what was going on with Alice lately. As Bruce Schneier noted in his Cato Institute keynote, "nobody here lies to their search engine." In our current system, it's relatively easy for surveilling entities to obtain court permission to track certain kinds of revealing activity because it's classified as "just metadata." The Columbia paper is due out next year. A recently passed digital privacy

law in California, traditionally a leader in these matters, also looks to an updated idea of what we mean by, and learn from, data that may not be so "meta" after all. An amicus brief filed with the Ninth Circuit Court in support of an appeal raised by Basaaly Saeed Moalin—the Somali man convicted in the sole successful Section 215 case mentioned above—is also apt to shape our discussion of metadata going forward. Jonathan Meyer, the FTC chief technologist mentioned above, has published on the topic as well.

The Internet of Things experienced significant negative attention for privacy weaknesses this year, and this will undoubtedly continue. Observers predict a great deal of pain as disparate industries attempt to harmonize their approaches to security and privacy, some of them very different from what the traditional tech community might expect or hope for. One potential solution involves minimizing the data sent by individual devices for processing in the cloud. This thought may be anathema to hardline cloud fans, but it would simply represent just another ebb and flow in the great cycle of client-server life. The legislative dam is expected to burst at any moment on drone regulation, though tech companies and would-be flyers are expressing frustration over Federal Aviation Administration (FAA) reluctance to tackle privacy implications of these craft.[189]

[178] http://www.law360.com/media/articles/724240.

[179] https://www.cs.columbia.edu/~smb/talks/ip-metadata-cato.pdf.

[180] http://www.cato.org/events/second-annual-cato-surveillance-conference.

[181] Op. cit.

[182] http://www.law360.com/privacy/articles/714875.

[183] http://www.law360.com/technology/articles/724382.

[184] http://www.law360.com/technology/articles/724382.

[185] https://jonathanmayer.org/.

[186] http://www.law360.com/privacy/articles/715752.

[187] https://iapp.org/news/a/can-data-minimization-be-the-answer-in-the-internet-of-things/.

[188] http://www.datamation.com/netsys/article.php/3865726/Trends-in-Thin-Client-Computing.htm.

[189] http://www.law360.com/privacy/articles/708682.

On the ground, expect more discussion on the rights of security researchers to poke at the inner workings of vehicles. Such legislation so far looks somewhat promising, because it would require auto manufacturers to have and publish privacy policies covering data collected by the car or shared by the driver, but many are concerned that other provisions in the legislation drafted so far would criminalize vehicle hacking[190]—especially after researchers in 2015 made it clear[191] that scrutiny is desperately needed. In other fields, 2015 saw the first instance in which the Federal Drug Administration (FDA) recommended discontinuing use of a medical device because of security concerns, but it is unlikely Hospira's situation will be the last of that kind.[192] It is hoped that greater familiarity will lead to better privacy practices for Fitbit Nation[193] and thousands of other users of applications transmitting personal medical data.[194] Developers of many types of mobile applications will find themselves making finer distinctions between "consumers" and "subscribers" to balance privacy rights and their need to get paid.[195] And the US adoption of chip-and-pin technology late in 2015 will provide good hunting for attackers willing to show us just how little we can trust the silicon around us.[196]

The world has in the past few years become more aware of the abuse tactic called "swatting," in which anonymous phone calls are made to summon highly armed police units to the homes of unwitting victims.[197] The legal situation around swatting has been murky, but it's generally understood that some sort of legal remedy is needed. It may take one or more very bad incident outcomes to raise swatting to the necessary level of public debate, but the odds are excellent that this will come to pass[198]—and with multiple big wins in 2015 against owners of "revenge porn" sites,[199, 200, 201] perhaps there's hope.

Fortunately, we can end this section on the brighter note—a hope, even, that the things we've learned in the online world can be made helpful both online and off. Bitcoin, that privacy-centered cryptocurrency, has gained new attention from government authorities in Honduras—not because of its financial prowess, but because Bitcoin architects figured out how to allow people who do not know or trust each other to collaborate on certain kinds of activity.[202] Both Honduras and Greece have expressed interest in using the blockchain concept at the heart of Bitcoin as a framework for handling land registries.[203]

[190] http://www.law360.com/privacy/articles/715319.

[191] http://www.wired.com/2015/07/hackers-remotely-kill-jeep-highway/.

[192] http://www.law360.com/privacy/articles/696325.

[193] http://www.nationaljournal.com/tech/2014/09/15/Fitbit-Hires-Lobbyists-After-Privacy-Controversy.

[194] http://www.law360.com/privacy/articles/710842.

[195] http://www.law360.com/technology/articles/730181.

[196] http://boingboing.net/2015/11/26/tiny-open-source-gadget-simula.html.

[197] http://www.nytimes.com/2015/11/29/magazine/the-serial-swatter.html.

[198] http://america.aljazeera.com/articles/2015/12/4/federal-bill-attempts-to-address-swatting-phenomenon.html.

[199] http://www.lawfuel.com/reprehensible-revenge-porn-site-operator-gets-jailed-email-hacking.

[200] http://www.christiantimes.com/article/revenge.porn.website.owner.gets.18.year.jail.sentence/51964.htm.

[201] http://www.dailymail.co.uk/news/article-2968929/Man-controversial-revenge-porn-site-demands-Google-remove-links-news-stories-critical-sleazy-empire-grounds-pictures-used-without-authorisation.html.

[202] http://www.economist.com/news/briefing/21677228-technology-behind-bitcoin-lets-people-who-do-not-know-or-trust-each-other-build-dependable.

[203] Op. cit.

Conclusion

Say what you will about privacy; there's always something interesting afoot. The Year of the Breach was followed by 2015's Year of Collateral Damage, as hacks exposing personal information of people with no direct relationship to the sites breached caused pain and mayhem for tens of thousands of innocent bystanders. The US federal government struggled with many privacy issues, even as the European Union and other entities pressed the accelerator on efforts to bring US companies in line with norms overseas. With geopolitical tensions worldwide as the year closed, it seems as if privacy issues will struggle in 2016 to keep their rightful footing side by side with security efforts.

Vulnerability methods, exploits, and malware

The past year saw a record number of advisories published by the ZDI.[204] While vendors continue to create patches to address individual bugs, efforts have also been taken to provide defenses for entire classes of vulnerabilities.

Take it to the source: vulnerability-specific mitigations

What happens when vulnerabilities are discovered? If everything works, patches are released. These patches typically comprise point fixes that remediate the discovered issue. It is a never-ending cycle of activities:

- Researcher uncovers zero-day vulnerability; reports to vendor.
- Developers implement a fix.
- Vendor releases a patch.
- End user deploys a software update.

All of this activity costs a significant amount of money and requires numerous man-hours to do correctly.[205] In the end, the user is secured from that vulnerability, which can no longer be used to breach a corporate network—at least until the next vulnerability is discovered. With code bases reaching millions of lines of code the next vulnerability is never far away.

What else can be done?

In the past, vendors analyzed exploits discovered in the wild and engineered countermeasures to combat the techniques used. Data execution prevention (DEP) and address space layout randomization (ASLR) are classic examples of these mitigations developed by vendors.

The DEP mitigation marks memory regions as executable or non-executable and denies data the option to be executed in non-executable regions. DEP can be enforced by hardware (when it is sometimes known as the NX [No-eXecute] bit) as well as by software. When released, it was a formidable defense against the standard exploitation techniques of executing shellcode from an attacker-controlled buffer. Today, however, attackers have several techniques to bypass DEP.

One common way is to use return-oriented programming (ROP) chains to call VirtualProtect/mprotect and flag a certain region as "executable." To combat this technique, vendors implemented ASLR to randomize the base address of loaded dynamic link libraries (DLLs), thus increasing the difficultly in fielding a reliable exploit. Attackers could rely on the known addresses of ROP gadgets to disable DEP. With the introduction of ASLR, attackers must either find a way to load a non-ASLR DLL or to leak a DLL address. With this new requirement for exploitation, vulnerabilities that disclosed the layout of memory in a process became highly prized in the attacker community.

These types of mitigations were successful at breaking the common exploit techniques of the time, but attackers worked their way around the defenses. The cat-and-mouse game between software vendors and exploit writers continues with new exploit-specific mitigations being released and new offensive techniques quickly following. While these mitigations evolved, new vulnerabilities continued to be discovered and patches deployed. Recently, vendors that receive hundreds of vulnerability reports began taking a different approach to software mitigations.

[204] http://www.zerodayinitiative.com/advisories/published/.

[205] http://blog.celerity.com/the-true-cost-of-a-software-bug.

New mitigation strategy

Microsoft's flagship browsers, Internet Explorer® and Edge®, offer a unique case study on this new approach to mitigations. Over the years, use-after-free (UAF) vulnerabilities became one of the most common vulnerability classes in the browser. They were the vulnerability of choice in everything from nation-state attacks to common campaigns launched from exploit kits.[206] In response to each one, Microsoft remediated and released patches for hundreds of UAFs. During this time, Microsoft not only implemented point fixes but also developed a new set of mitigations hoping to eliminate this vulnerability class.[207]

In the summer of 2014, Microsoft introduced two new mitigations into its browser to increase the complexity of successfully exploiting a UAF. June's patch[208] introduced a separate heap, called isolated heap, which handles most of the document object model (DOM) and supporting objects. From a defensive perspective, the isolated heaps make it harder for an attacker to fill a freed object residing inside the isolate heap region with controlled values.

Microsoft released a subsequent patch[209] in July, 2014, introducing a new strategy for freeing memory on the heap—MemoryProtection. This mitigation operates by preventing memory blocks from being deallocated as long as they are being referenced directly on the stack or processor registers. MemoryProtection guarantees the block will remain on the wait list until reuse, and will remain filled with zeroes. This prevents an attacker from controlling the contents of the freed block before it is reused. Both of these mitigations had an immediate impact on the use-after-free landscape by implementing techniques to mitigate the effects of the vulnerability's existence.

Isolated heap and MemoryProtection were not the only use-after-free mitigations in the development pipeline at Microsoft. MemGC was introduced in Microsoft Edge and Internet Explorer browsers in Windows® 10.[210] MemGC is a major evolutionary step, improving upon the protections afforded by MemoryProtection. In October of 2015,[211] MemGC was additionally back-ported to Internet Explorer 11 running on earlier Windows versions.

MemoryProtection only guards against references to freed objects residing on the stack or processor registers. In contrast, MemGC aims to provide protection to a full-fledged managed memory solution by protecting against references to freed objects regardless of where the reference may live.

MemGC knows of all allocations made through the MemGC allocator. When application code requests to free an allocated block of memory, MemGC fills the memory with zeroes. This serves as an effective mitigation against UAFs. MemGC keeps the memory in an allocated and zeroed state. Periodically, MemGC executes a "recycling" operation to perform final deallocation of all such memory blocks. A memory block will only be recycled when no references remain, either on stacks or in other MemGC-tracked allocations.

MemGC represents a highly effective mitigation. At the time of this writing, the vast majority of use-after-free vulnerabilities in Microsoft Edge and Internet Explorer 11 are rendered non-exploitable by this mitigation.

New industry norms

For complex code bases like web browsers, mitigations targeting the effectiveness of a common vulnerability type are a welcome change. Fortunately, Microsoft is not the only vendor developing these types of countermeasures. Mozilla Firefox implemented Frame Poisoning[212] and Google Chrome developed PartitionAlloc[213] to combat use-after-free vulnerabilities. These mitigations increase the complexity of successfully writing a reliable exploit leveraging this style of vulnerability. The browser developers successfully disrupted the threat landscape and forced attackers to adjust their tactics, which is the ultimate goal.

[206] http://community.hpe.com/t5/Security-Research/Microsoft-IE-zero-day-and-recent-exploitation-trends-CVE-2014/ba-p/6461820#.VnQmo_mDFBc.

[207] https://securityintelligence.com/understanding-ies-new-exploit-mitigations-the-memory-protector-and-the-isolated-heap/.

[208] https://technet.microsoft.com/en-us/library/security/ms14-035.aspx.

[209] https://technet.microsoft.com/en-us/library/security/ms14-037.aspx.

[210] https://blogs.windows.com/msedgedev/2015/05/11/microsoft-edge-building-a-safer-browser/.

[211] https://technet.microsoft.com/en-us/library/security/ms15-106.aspx.

[212] https://bugzilla.mozilla.org/show_bug.cgi?id=497495.

[213] http://blog.chromium.org/2014/08/64-bits-of-awesome-64-bit-windows_26.html.

Logical abuses of implicit calls

There is no denying 2015 was the year for active exploitation of Adobe Flash. This was driven by the existence of an easy-to-use exploit primitive made available through the corruption of vector objects and an abundance of UAFs in the code base. Given all this attention, Adobe Flash is being heavily audited by security researchers interested in aiding in the fight against adversaries,[214] but it is not the only Adobe product receiving attention. Adobe Reader fixed a record number of vulnerabilities[215] in the 2015 calendar year.

Most of the Reader vulnerabilities discovered reside in the code handling the JavaScript APIs. These JavaScript APIs offer document authors a rich set of functionality, allowing them to process forms, control multimedia events, and communicate with databases. The primary purpose for this flexibility is to give the end user easy-to-use yet complex documents. Unfortunately, this flexibility is a perfect avenue for attackers. By thinking outside the box, an attacker can execute malicious logic by leveraging weaknesses in this code base.

Adobe built a security boundary into these APIs.[216] The boundary limits what type of functionality is made available to document authors based on the mode in which the application is operating. In Adobe Reader, this security boundary is implemented based on the concept of privileged and non-privileged context. When the code is executing in a privileged context, the document author is allowed access to the subset of security-restricted APIs. Examples of points within Adobe Reader where code executes in a privileged context include operating in console mode, performing batch operations, executing application initialization events, and trusting the document's certificate. During these times, the document author will have access to the security-restricted APIs.

Some examples of non-privileged APIs include mouse-up and mouse-down events and any functionality that can be executed in the "doc" context. A specific example of a security-restricted API is app.launchURL. This API should not be available from the "doc" context and should only be executed when you are in batch or console mode. If you try to execute a privileged API from the doc context, you will be prompted with the following error dialog:

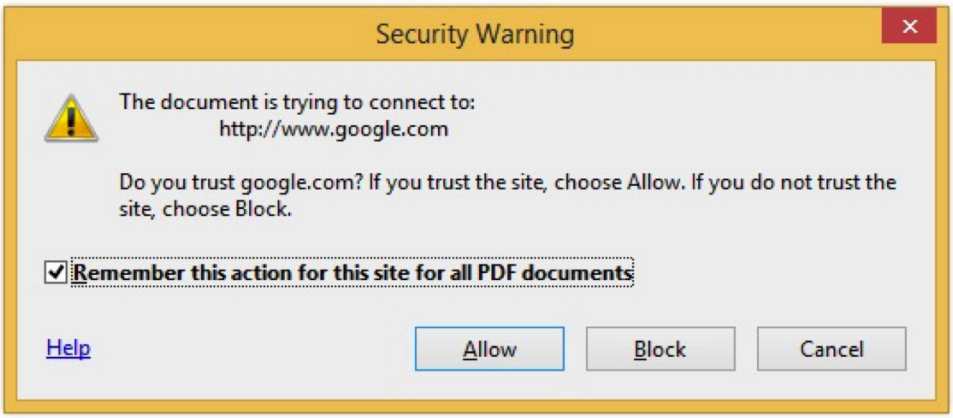


Figure 9. Adobe Reader security warning

An attacker's goal is to execute a privileged API (or security-restricted function) from within the "doc" context, providing the ability to execute unintended operations by strictly viewing the PDF. This needs to be completed without alerting the victim with a security warning dialog. Surprisingly, attackers do not need to resort to classic memory corruption techniques to accomplish the goal. This can be accomplished by simply understanding when the JavaScript language makes implicit function calls. These implicit calls allow the attacker to execute user-defined code in an unintended context.

[214] http://krebsonsecurity.com/tag/adobe-flash-player/.

[215] https://helpx.adobe.com/security.html.

[216] https://blogs.adobe.com/security/2010/12/leveraging-the-android-sandbox-with-adobe-reader.html.

Using property redefinition techniques, the attacker gains the ability to execute arbitrary security-restricted APIs from a context in which they are not allowed. Now all the attacker needs to do is find some interesting security-restricted APIs to call. In this case, Adobe Reader's undocumented APIs[217] fulfill this exact requirement. The undocumented privileged API Collab.uriPutData provides the end user the ability to dump a file to disk. Using the undocumented API, attackers can either stash their payload in the victim's startup folder or drop a DLL in the disk. Either option allows them to gain remote code execution of the victim's machine.

With these exploit primitives, attackers have everything needed to construct an exploit that achieves remote code execution through JavaScript API restriction bypass vulnerabilities. They begin their attack by attaching a malicious payload to a PDF. Next, they write JavaScript that executes when the document is opened. The JavaScript needs to extract the contents of the attachment into a JavaScript object. Following that, they leverage a JavaScript API restriction bypass vulnerability to execute the undocumented privileged API Collab.uriPutData to drop a DLL to the disk. Once the DLL is dropped, they force Adobe Reader to load the attacker-supplied DLL and execute the payload.

This type of attack is devastating as there is little indication of compromise. In this case, the attacker is not corrupting memory within the application so the application will not crash. No security dialog will be displayed to the end user to show that privileged functionality is being executed. The exploit is simply redefining what methods are implicitly called within the JavaScript so that it executes unintended privileged APIs. These abuses offer an interesting case study of when using the language as designed results in a security boundary failure.

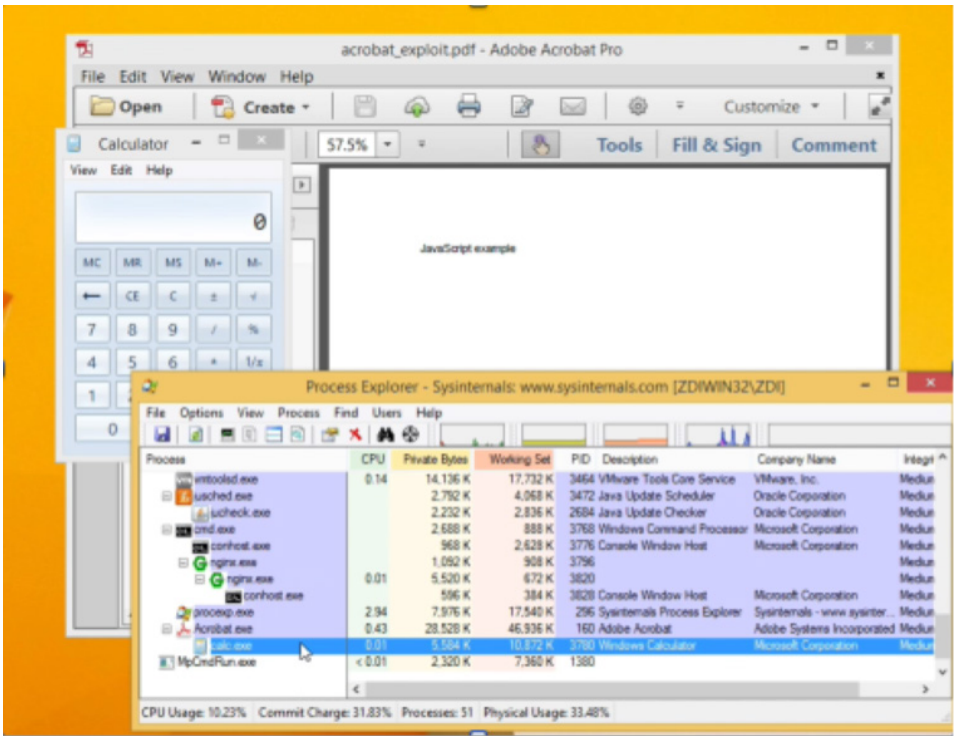


Figure 10. The result of successful exploitation

[217] http://www.adobe.com/content/dam/Adobe/en/devnet/acrobat/pdfs/js_api_reference.pdf.

The need for wide-reaching fixes

While the fixes for use-after-free vulnerabilities in Microsoft Internet Explorer and Edge are commendable, history teaches us it is only a matter of time before attackers leverage a different vector to exploit these programs. Still, the inclusion of MemoryProtection and MemGC demonstrates how wide-reaching fixes disrupt attack in an asymmetric fashion. Instead of releasing patches to fix many different vulnerabilities, these defensive measures take out the entire class—at least for some period of time. Other vendors would do well to consider implementing similar strategies to disrupt classes of attacks.

Exploits

As detailed in the "business of bugs" section earlier in this report, finding vulnerabilities in software is usually the domain of security researchers, with many of them participating in coordinated disclosure with vendors. Despite the progress made with a record-setting year (both reporting and patching), exploits remained one of the main vectors allowing remote code execution and privilege escalation by attackers.

The distribution of newly discovered samples for vulnerabilities identified in 2015 (CVE-2015-xxxx) shows the high prevalence of exploits for the Windows privilege escalation vulnerability CVE-2015-1701,[218] which accounts for over 45% of exploit samples for the year. CVE-2015-1701, first observed in a highly targeted attack,[219] was used in combination with Adobe Flash remote code execution exploits.

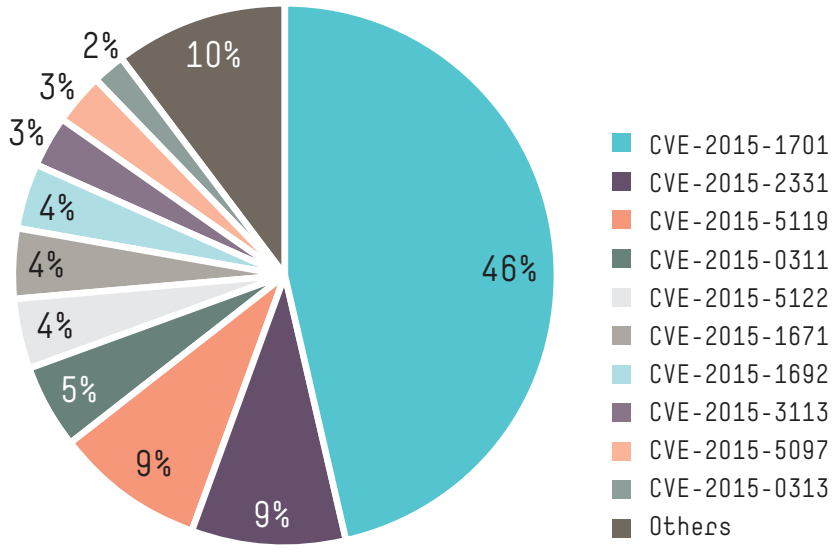


Figure 11. Top 10 CVE-2015 exploits by prevalence discovered by ReversingLabs

Looking at vulnerable applications (Figure 12), however, the top 20 were dominated by Adobe Flash exploits. Indeed, 2015 was fraught with newly discovered Flash vulnerabilities in spite of several security improvements implemented by Adobe and Microsoft Windows such as Control Flow Guard (CFG) and a more secure Action Script vector class. Out of the top 20 applications, half affected Adobe Flash. The most commonly encountered Flash samples include two discovered after the Hacking Team breach (CVE-2015-5119[220] and CVE-2015-5122[221]), and a third (CVE-2015-0311[222]) found in the Angler exploit toolkit.[223]

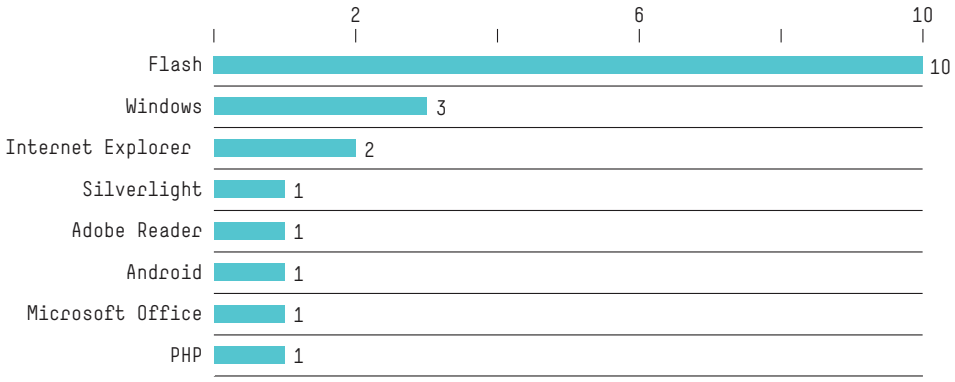


Figure 12. Top 20 vulnerabilities by targeted platform

Looking at the proportion of samples discovered by ReversingLabs in 2015, it is Microsoft vulnerabilities that dominate the top 20, accounting for nearly 50% of all discovered samples, followed by Adobe Flash (29%) and a single PHP vulnerability (10%).

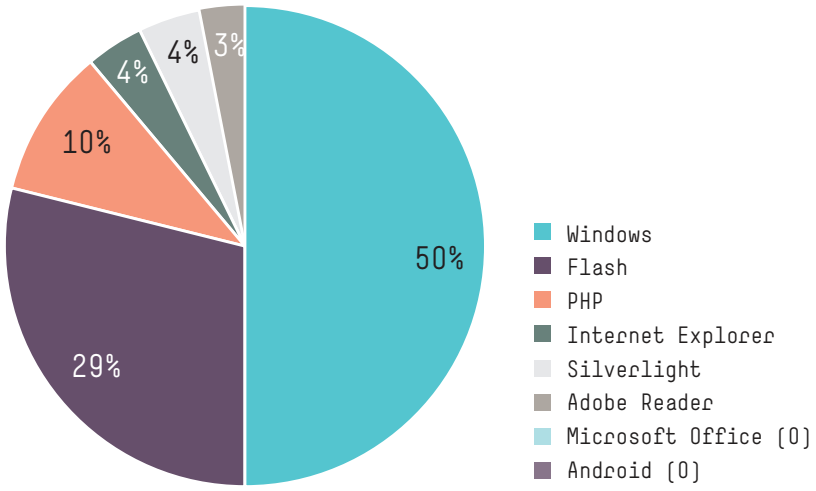


Figure 13. Proportion of CVE-2015-xxxx vulnerabilities discovered by affected application

[218] http://www.cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-1701.

[219] https://www.fireeye.com/blog/threat-research/2015/04/probable_apt28_useo.html.

[220] https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-5119.

[221] https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-5122.

[222] https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-0311.

[223] http://malware.dontneedcoffee.com/2015/01/unpatched-vulnerability-0day-in-flash.html.

As we learned in last year's report, attackers leverage more than just the newest vulnerabilities to carry out successful attacks. Looking at newly discovered exploit samples for all known vulnerabilities, not just those discovered in 2015, different patterns emerge. Alarmingly, 2015 is still dominated by CVE-2010-2568[224] (patched again in early 2015[225]), although the overall proportion is a bit lower (29% in 2015 vs. 33%in 2014). In fact, it is disheartening to see that the top 10 vulnerabilities exploited overall (Figure 14) continue to be those that are more than a year old (and 48% are five or more years old).

Surprisingly, the old Shortcut Icon Loading Vulnerability first discovered by VirusBlokAda during the initial analysis of Stuxnet is followed by samples exploiting CVE-2012-6422,[226] a vulnerability in Samsung Exynos processors, which allows the attacker to escalate privileges by being able to arbitrarily read and write system memory. The prevalence of CVE-2012-6422 is likely indicative of the increased popularity of Samsung smartphones and the Android operating system.

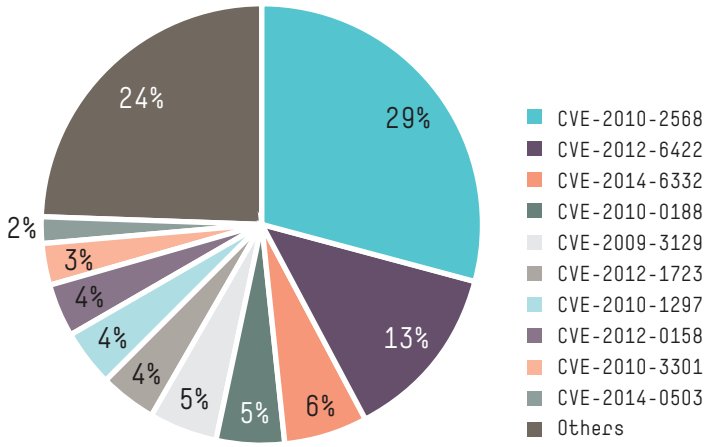


Figure 14. Top vulnerabilities exploited in 2015 by as reported by ReversingLabs

Now, let's look at this from the perspective of targeted applications and platforms. Again, the data shows Microsoft Windows dominates with more than 42%—largely attributed to the Stuxnet vulnerability (CVE-2010-2568). Confirming our belief that attackers are very much interested in mobile platforms, Android is second with 18%. Oracle Java (12%), Microsoft Office (11%), and Adobe (Flash and Reader at 7% each) round out the top five.

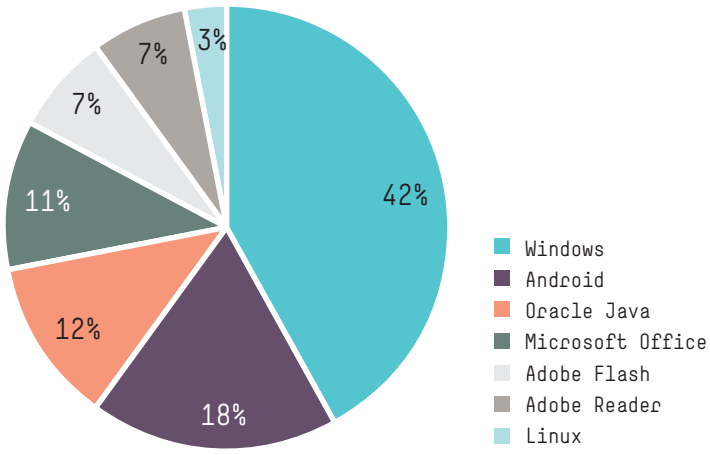


Figure 15. Top 20 discovered exploit samples by targeted platform

[224] https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2010-2568.

[225] http://community.hpe.com/t5/Security-Research/Full-details-on-CVE-2015-0096-and-the-failed-MS10-046-Stuxnet/ba-p/6718459.

[226] https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2012-6422.

With the exception of Windows and Android, the platforms represented here are used most commonly for delivering malicious exploit files resulting in malware infections. This is a change from last year's report when Java exploits were the second most prevalent, accounting for more than 21% of all discovered exploit samples.

Although several vulnerabilities in JRE were discovered in 2015, none of them allowed remote code execution,[227] which lowers the interest of malware attackers in Java. Combine this with the fact that many people learned how to disable Java from running within a web browser[228] environment, and it is easy to understand why Java fell in 2015—at least as a platform.

Looking at only newly discovered exploit samples delivered through web pages, the situation is somewhat different as HTML (JavaScript) leads with 35%—more than Adobe Reader, Java files, and Adobe Flash files.

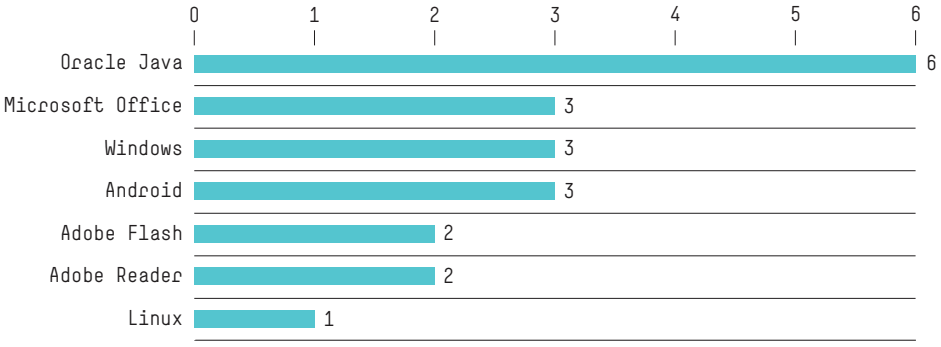


Figure 16. Count of newly discovered exploit samples by platform

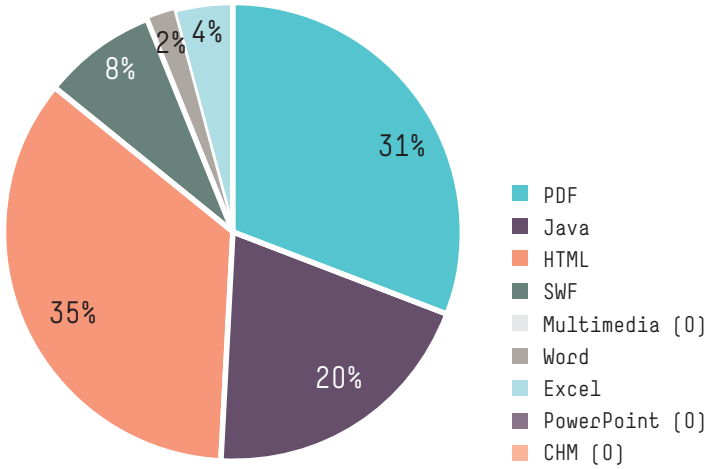


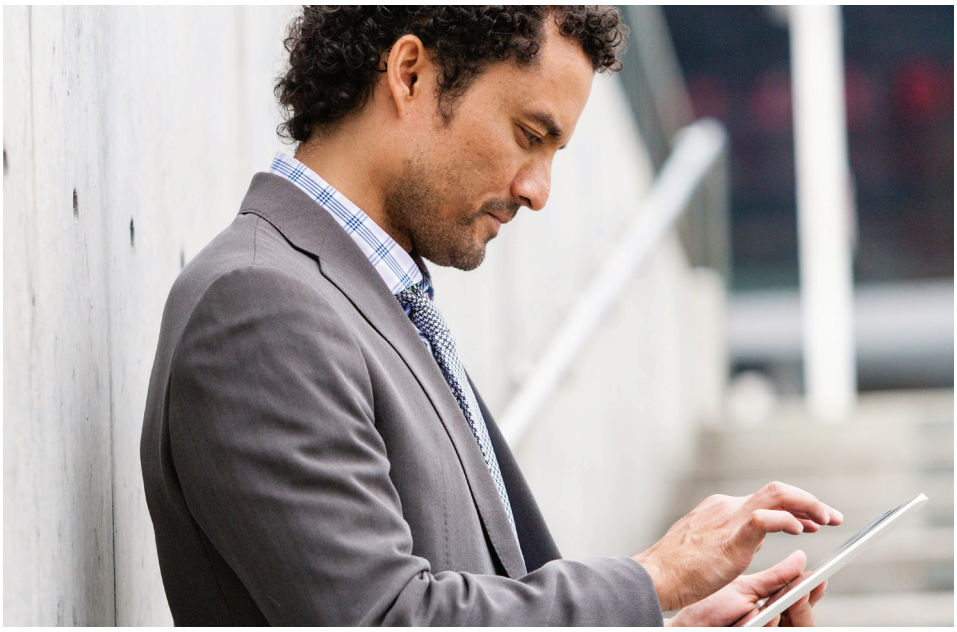
Figure 17. Web or email exploit samples discovered in 2015 by file type

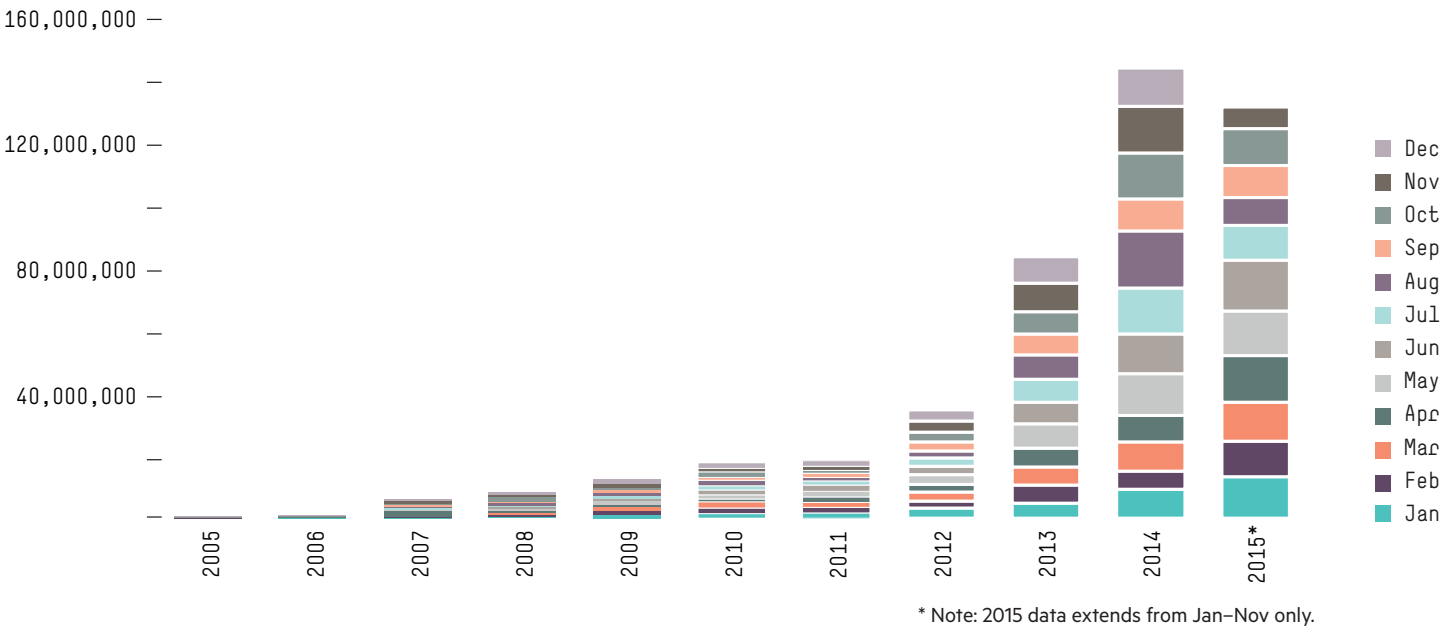
[227] http://www.oracle.com/technetwork/topics/security/alerts-086861.html.

[228] http://java.com/en/download/help/disable_browser.xml.

Malware: still dangerous, still pervasive

While malware still represents a significant threat to the digital enterprise in 2015, it seems the linear growth of newly discovered malware samples did not materialize. According to the independent German security testing organization AV-TEST, there were about 140 million newly discovered malware samples for the Windows platform.[229] This largely agrees with the 135 million samples HPE Security Research partner ReversingLabs tracked. It's important to note that ReversingLabs' number includes potentially unwanted applications (PUAs).





* Note: 2015 data extends from Jan–Nov only.

Figure 18. Newly discovered samples in AV-TEST repository

[229] https://www.av-test.org/en/statistics/malware/.

Given trending over recent years, the expectation was that it would continue at the same alarming pace into 2015. Even we predicted this in our 2015 report. It is difficult to pinpoint exact reasons for this apparent stagnation, but it's likely due to improvements in defenses such as the operating system and protection components implemented in the enterprise. Another contributing factor could be the takedown of several large malware operations in 2015.[230, 231] We can also reasonably attribute some percentage of decline to the consumer shift from traditional computers to mobile devices. The centralized distribution model for apps used by iOS and Android has proven more difficult for malware attackers despite the obvious growth in interest in attacking mobile platforms.

Regardless of increased interest in attacking mobile platforms, Microsoft Windows remains the top platform for malware with 94%, as seen in Figure 19.

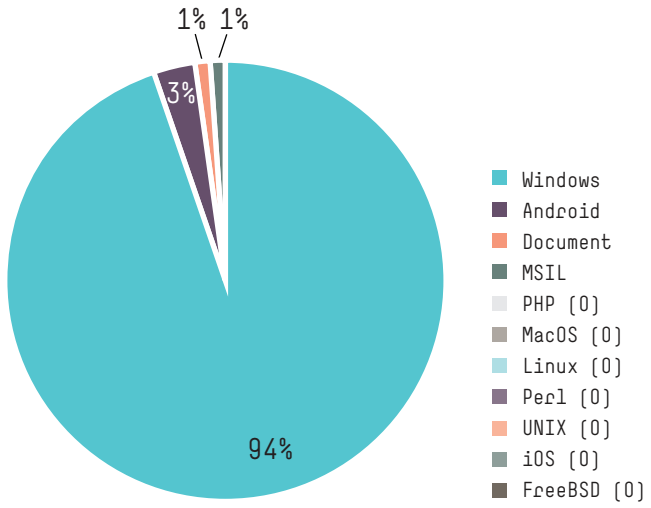


Figure 19. Breakdown of malware samples by platform discovered in 2015 by ReversingLabs

[230] http://www.consumerreports.org/cro/news/2015/04/botnet-takedown-removes-malware-threats/index.htm.

[231] https://threatpost.com/law-enforcement-shuts-down-dridex-operation/115036/.

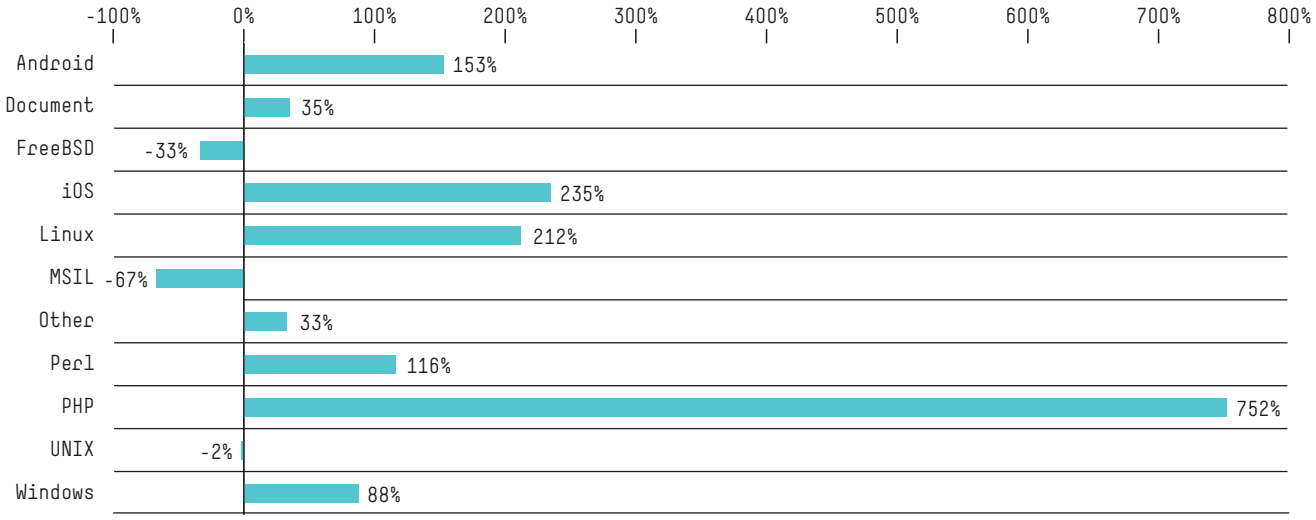


Figure 20. Yearly growth in newly discovered malware samples by platform (ReversingLabs)

The only other platform of note here is Android, with 3% (or just over 4.5 million samples). Of interest, Android's (and other mobile platforms') main threats are potentially unwanted applications (PUAs) and advertising frameworks collecting private and potentially identifiable user information.

Examining overall growth rates per platform, we see a shift away from Windows-only malware. The absolute champion, in terms of growth rate, was Apple iOS, with an increase of more than 230% (although the number of discovered Apple malware samples, just under 70,000, is still very small compared to the number of Windows malware samples) as seen in Figure 20.

The increase of interest in Linux can be contributed to its popularity for hosting web content.[232] Users of popular content management applications such as WordPress[233] often fail to promptly update to the latest versions, allowing attackers time to install malicious code on the server. The growth of PHP-based samples can be attributed to remote administration (remote shell) tools exposed through a web-based user interface.

Overall, despite being a very serious problem, we can be happy with a slowdown in Windows-based malware—at least for 2015. Security features and mitigations as addressed earlier in this report are expected to further contribute to a stagnation in the growth of malware and hopefully, an eventual decline.

[232] http://w3techs.com/technologies/overview/operating_system/all.

[233] http://w3techs.com/technologies/overview/content_management/all.

Windows malware in 2015

Looking more closely at Windows malware, the data reveals that self-replicating malware such as network worms and parasitic viruses dominate. However, most of the top families are not new, showing us that patching is still very much a problem. Let's look at the top two families (Figure 21) more closely.

Allaple[234] tops the charts with 26% of samples and is a polymorphic worm discovered more than eight years ago that affects HTML files, which may contribute to such a high number of samples. Allaple is similar to Ramnit (infecting files other than Windows PE files), which accounted for almost 5% of the samples despite being taken down in February by Europol.[235]

Elkern,[236] at 19%, is another surprise. Elkern is a polymorphic parasitic virus and worm discovered in the old era of viruses (over 10 years ago) when malware was created to showcase the author's technical skills or, at worst, overwrite or delete files.

Although the growth in newly discovered malware samples slowed, 2015 was not without innovation including malware designed to attack users in general, as well as malware designed to target specific organizations.[237]

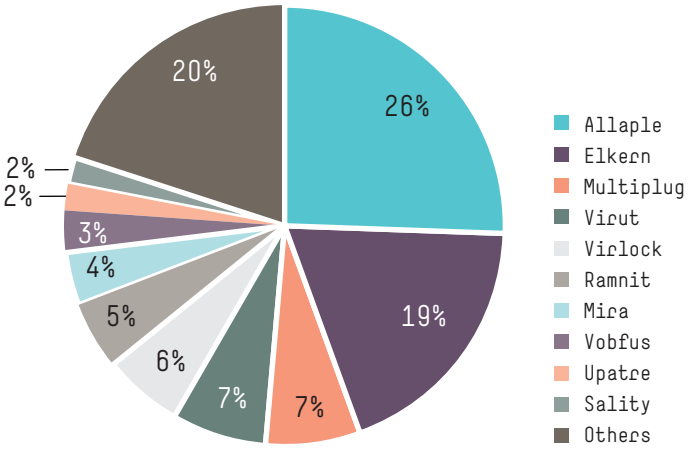


Figure 21. Top discovered Windows malware families according to ReversingLabs

[234] http://www.microsoft.com/security/portal/threat/encyclopedia/entry.aspx?Name=Win32%2fAllaple.

[235] https://nakedsecurity.sophos.com/2015/02/27/europol-takedown-of-ramnit-botnet-frees-3-2-million-pcs-from-cybercriminals-grasp/.

[236] http://www.microsoft.com/security/portal/threat/encyclopedia/entry.aspx?Name=Win32%2FElkern.

[237] https://www.anthemfacts.com/

OS X malware in 2015

2015 saw the continued trend of a relatively few truly malicious samples created to run specifically on OS X. Perhaps the most interesting examples are Bitcoin stealers such as CoinThief[238] or Bitcoin mining malware, which uses a computer's computing resources to mine Bitcoins for the benefit of the attacker.

The majority of threats on OS X, in fact over 99% of all newly discovered threats, belong to the category of potentially unwanted applications (PUAs). They are usually installed in a bundle along with useful, wanted applications.[239] Once installed, PUAs often download and install additional components or display advertisements through browser plugins such as VSearch.[240] The most commonly encountered OS X PUA is known as Macnist,[241] which encompasses several downloader families.

Although the malware protection module Gatekeeper, built into the OS X operating system, is improving with every new release, this year has seen a few successful attacks designed to bypass it.[242] We expect that the research in this area will continue in the next year, as bypassing built-in security mechanisms is key to successfully installing malicious software on OS X.

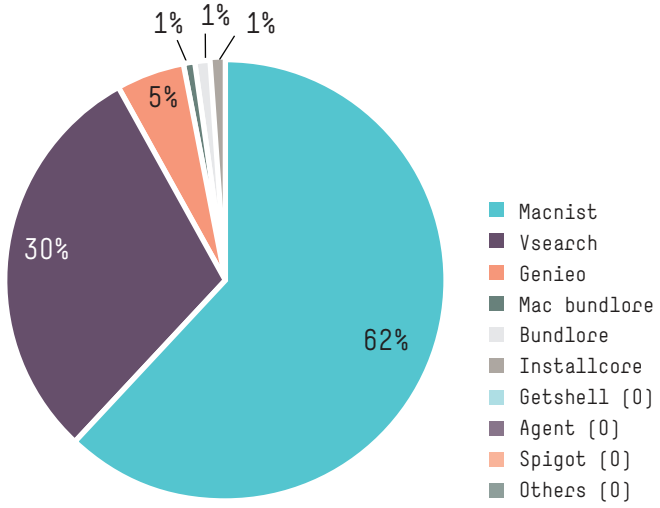


Figure 22. Top OS X threat samples discovered in 2015

[238] http://www.securemac.com/privacyscan/new-apple-mac-trojan-called-osxcointhief-discovered.

[239] https://malwaretips.com/blogs/remove-potentially-unwanted-program/.

[240] http://www.thesafemac.com/arg-vsearch/.

[241] https://cynomix3.appspot.com/tag/not-a-virus%3AHEUR%3ADownloader.OSX.Macnist.a.

[242] https://blog.malwarebytes.org/mac/2015/10/bypassing-apples-gatekeeper/.

Linux malware in 2015

Top Linux malware samples detected in 2015 are still dedicated to launching distributed denial of service (DDoS) attacks. Most DDoS Trojans connect to a central command and control (C&C) server used by the attackers to synchronize denial of service attacks, usually conducted by choosing one of the well-known[243] DoS methods such as TCP, UDP, and ICMP flood or DNS amplification.

The world of Linux malware, in terms of threat types, has not seen a major change from the last year and DDoS malware continues to dominate the top 20 threats, accounting for more than 60% of all discovered samples. A notable trend is the further increase in infecting small office and home routers, which are often misconfigured or not updated with the latest security patches. Several Linux malware families spread in this manner with functionally identical executables designed to run on different processor architecture, with MISP, ARM, x86, and PowerPC being the most common ones.

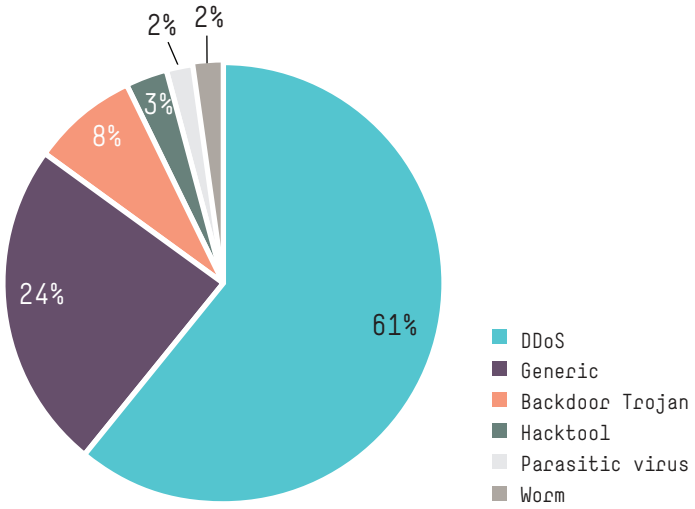


Figure 24. Top 20 Linux malware families by malware type

[243] http://www.techrepublic.com/blog/it-security/ddos-attack-methods-and-how-to-prevent-or-mitigate-them/.

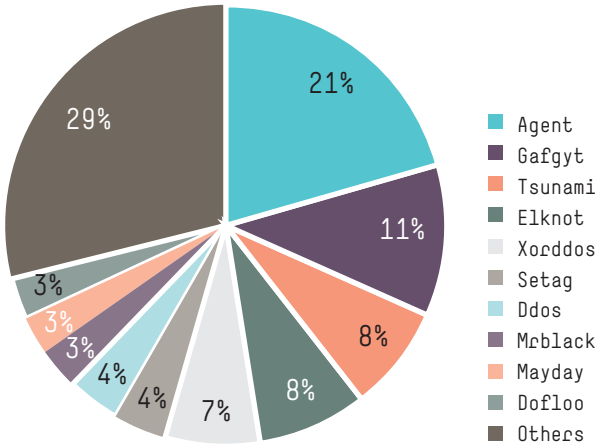


Figure 23. Top Linux malware samples discovered in 2015

First discovered in 2014, the Linux malware families Darlloz and Aidra[244] continued to target routers throughout 2015. Samples of both families attempted to secure infected devices from infection by another family, reminiscent of the Netsky, Bagle, and MyDoom wars,[245] which happened over 10 years ago.

This year, router-based malware has seen the addition of the purportedly "benevolent" worm Wifatch,[246] which spread to unprotected routers in an attempt to secure them from further infections with no malicious payloads included in its code. First mention of benevolent viruses was recorded in one of the first-ever research efforts into computer viruses, published by Fred Cohen in 1984. In his book "Computer Viruses—Theory and Experiments,"[247] Cohen describes a compression virus,[248] which preserves the original function of the infected executables but also saves disk space by compressing the host code.

The level of sophistication in Linux malware is still generally low, which can be attributed to the relatively low level of user awareness about malware threats to Linux. An additional contributing factor is the fact that anti-virus software is rarely installed on Linux systems. More advanced anti-debugging and obfuscation techniques are generally not used as they are typically not required when infecting systems.[249]

With the further increase of Linux as a popular platform for hosting applications and the adoption of software containers as a de facto standard for packaging applications, it is likely we will see more sophisticated Linux malware in the near future.

[244] http://now.avg.com/war-of-the-worms/.

[245] http://news.bbc.co.uk/2/hi/technology/3532009.stm.

[246] http://www.darkreading.com/vulnerabilities---threats/and-now-a-malware-tool-that-has-your-back/d/d-id/1322451.

[247] https://web.eecs.umich.edu/~aprakash/eecs588/handouts/cohen-viruses.html.

[248] http://all.net/books/virus/part2.html.

[249] http://www.zdnet.com/article/two-stealthy-linux-malware-samples-uncovered-following-in-windows-variants-tracks/.

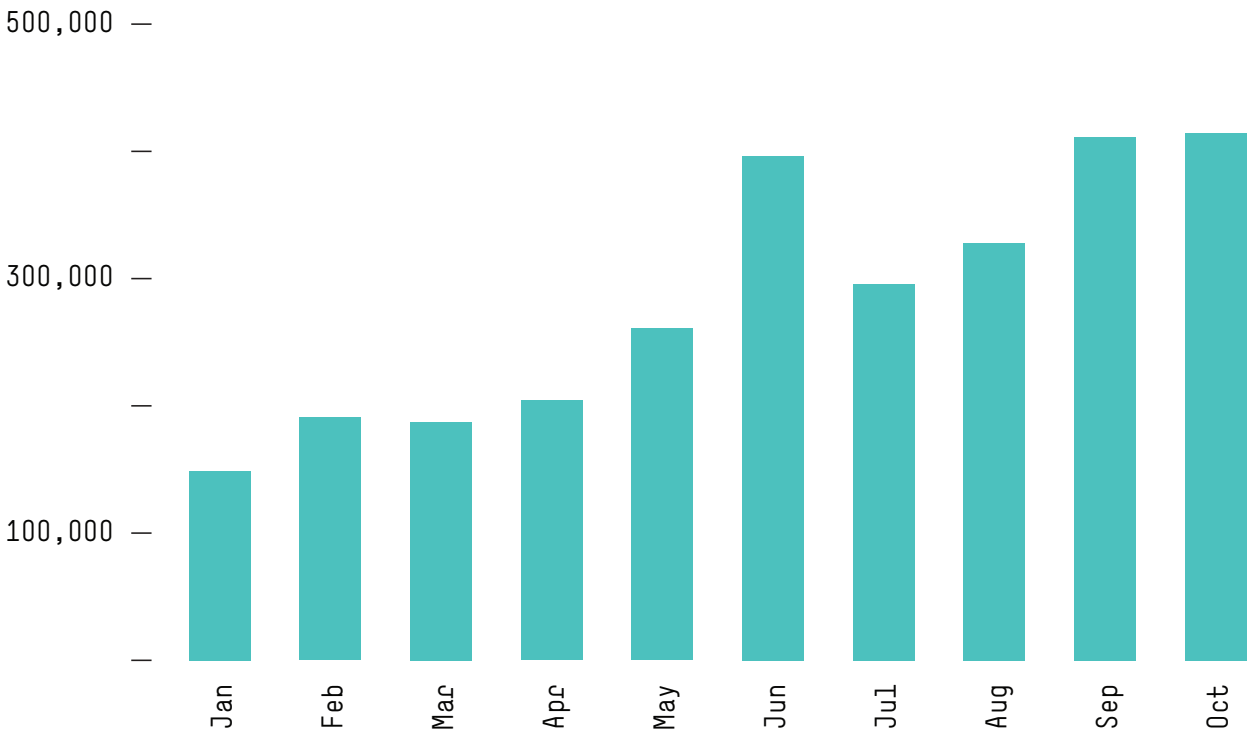


Figure 25. Discovered Android threats, through October 2015

Mobile malware in 2015

Android threats, malware, and potentially unwanted applications continued the growth trend already visible in 2014. Although there are nowhere near the number of discovered threats for Windows platform, we have reached the point where we are seeing over 10,000 new threats discovered daily, reaching a total year-over-year increase of 153%.

All the top malware families are the usual suspects in the Android world and are designed to obtain financial benefits for the attackers by installing additional unwanted components, sending or stealing SMS messages, or stealing confidential information such as the user's contacts details or phone's unique identifier. Most Android malware purports to be legitimate apps uploaded on third-party app repositories following patterns well-known from desktop malware.[250]

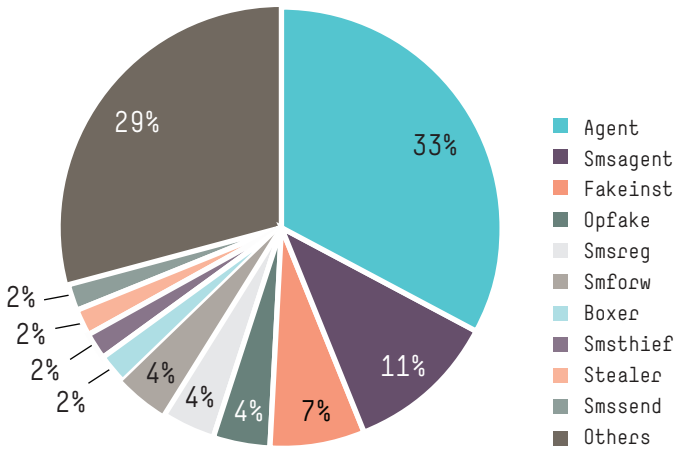


Figure 26. Top Android malware families discovered by ReversingLabs in 2015

[250] http://www.techtimes.com/articles/104373/20151109/new-family-of-android-malware-virtually-impossible-to-remove-say-hello-to-shedun-shuanet-and-shiftybug.htm.

Android ransomware

Ransomware is a very successful model of attack and its mobile variant is not much different from its desktop counterpart. Usually, the user is tricked into installing a useful app—for example, an app that pretends to be Adobe Flash player.[251] Once installed and executed, the malicious application attempts to encrypt all accessible documents, images, and multimedia files on the device. When this process is finished, the ransomware application displays a text, a warning that often seems to come from law enforcement agencies such as the FBI[252] and instructs the user how to pay to restore files and access to the device.

Some of the most successful Android ransomware families are Simplocker[253] and Koler.[254] The recently discovered Locker[255] family actually sets a PIN for the device and makes the restore almost impossible if the user is not willing to pay the attackers for recovery instructions.

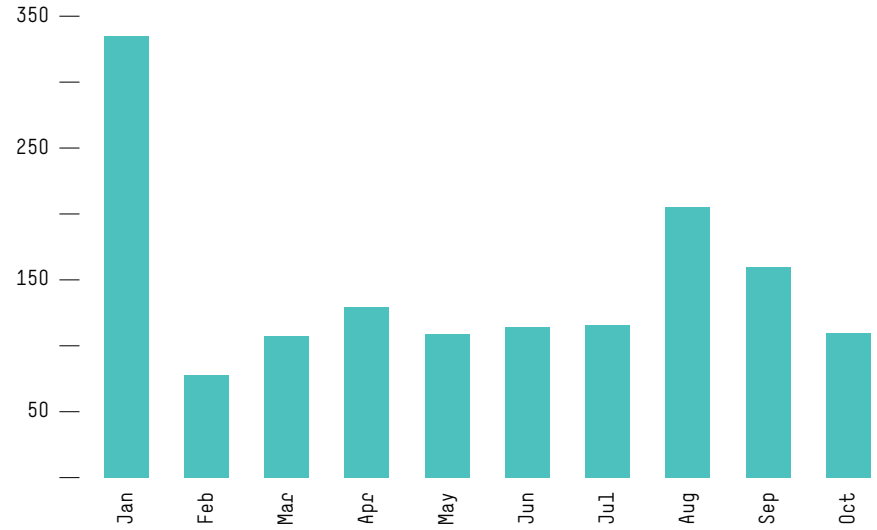


Figure 27. Monthly breakdown of Android ransomware samples discovered in 2015

Banking (phishing) malware

Fake Internet banking apps are another successful attack pattern that uses purported banking apps to steal user credentials and allow attackers access to user's bank accounts. The fake banking app attacks often targeted Korean,[256] Russian, Indian, or Vietnamese banks.[257] In Korea, users received spoofed SMS messages that enticed them to install the fake app, which led to loss of banking credentials. In other cases, such as those involving the Zeus malware,[258] fake apps required users to enter a code to access online banking while forwarding the access code to the attacker by sending a background SMS message.

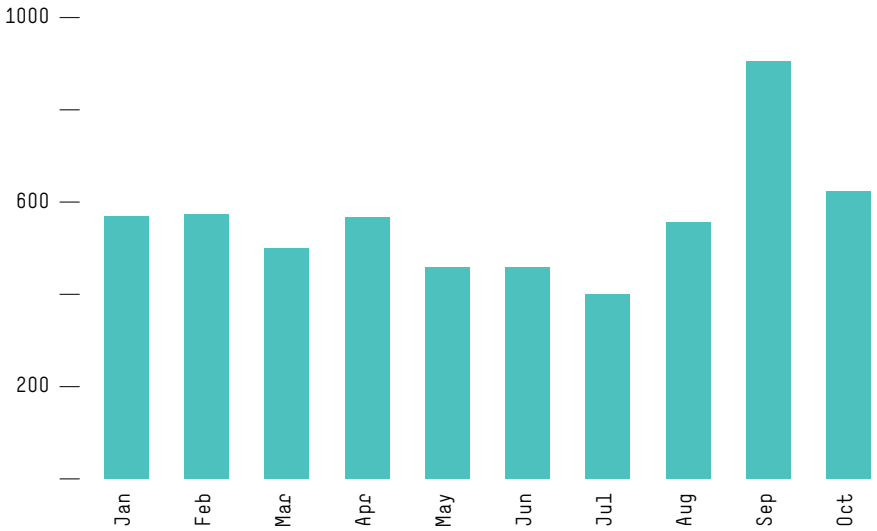


Figure 28. Monthly breakdown of fake banking apps malware discovered in 2015

[251] https://blog.avast.com/2015/02/10/mobile-crypto-ransomware-simplocker-now-on-steroids/.

[252] http://www.theregister.co.uk/2015/05/26/android_ransomware_mobile_scam_fbi/.

[253] http://www.welivesecurity.com/2014/07/22/androidsimplocker/.

[254] https://blog.malwarebytes.org/mobile-2/2014/05/difficulty-removing-koler-trojan-or-other-ransomware-on-android/.

[255] http://www.welivesecurity.com/2015/09/10/aggressive-android-ransomware-spreading-in-the-usa/.

[256] http://www.trendmicro.com/cloud-content/us/pdfs/security-intelligence/white-papers/wp-the-south-korean-fake-banking-app-scam.pdf.

[257] https://blog.kaspersky.com/android-banking-trojans/9897/.

[258] http://www.wiki-security.com/wiki/Parasite/ZeusTrojan/.

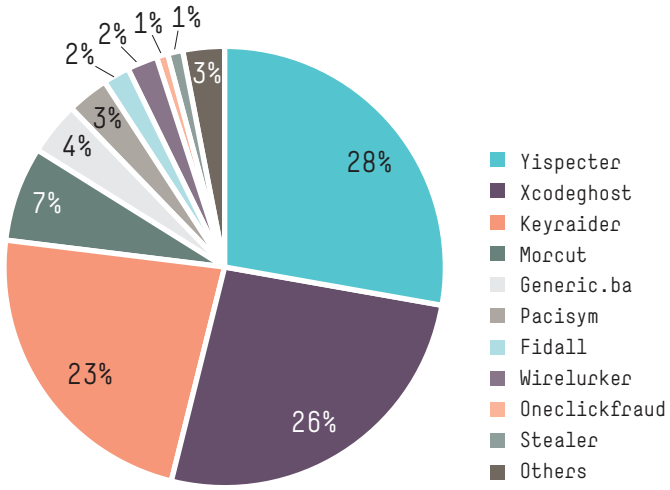


Figure 29. Top malicious iOS apps discovered in 2015

Information-stealing malware

Phone numbers, email addresses, and other contact details are valuable for malware writers. Many malicious applications, including PUAs, will attempt to collect this data from infected devices and send it to attackers. Apart from sending SMS messages to invite users to install malicious apps, some attackers also spread them through Twitter and WhatsApp messages.[259] In Japan, a group behind the Godwon information stealing malware collected the contact details to try to extort the user[260] after sharing compromising photos and videos with attackers.

iOS malware

Due to the popularity of the platform, researchers have been investigating all factors of the security of the iOS platform since its release. However, until 2015, we have not seen any successful malware attacks get into Apple's closely guarded App Store.

Unfortunately, this year witnessed the first major compromise of applications uploaded to the App Store. This is a consequence of an attacker's modification of Apple's Xcode[261] programming environment, which was shared among many developers in China.[262] This resulted in a malicious information-stealing component, known as XcodeGhost,[263] being included with more than 4000 apps[264] published to the App Store by legitimate developers of IOS apps.

While XcodeGhost managed to get into Apple's App Store, two additional families targeting third-party app stores for jailbroken phones became prevalent: Yispecter[265] and Keyraider.[266] Interestingly all major iOS malware families are geographically targeting Chinese and Taiwanese users.[267]

Although the total number of IOS malicious apps is very low compared to all other popular malware platforms, the growth of 235% indicates that it should be a closely watched area in 2016.

[259] https://blog.lookout.com/blog/2015/01/06/socialpath/.

[260] http://www.symantec.com/connect/blogs/online-criminal-group-uses-android-app-sextortion.

[261] https://developer.apple.com/xcode/.

[262] http://techcrunch.com/2015/09/21/apple-confirms-malware-infected-apps-found-and-removed-from-its-chinese-app-store/.

[263] http://researchcenter.paloaltonetworks.com/2015/09/novel-malware-xcodeghost-modifies-xcode-infects-apple-ios-apps-and-hits-app-store/.

[264] https://www.fireeye.com/blog/executive-perspective/2015/09/protecting_our_custo.html.

[265] http://researchcenter.paloaltonetworks.com/2015/10/yispecter-first-ios-malware-attacks-non-jailbroken-ios-devices-by-abusing-private-apis/.

[266] http://researchcenter.paloaltonetworks.com/2015/08/keyraider-ios-malware-steals-over-225000-apple-accounts-to-create-free-app-utopia/.

[267] http://www.macrumors.com/2015/09/20/xcodeghost-chinese-malware-faq/.

Spotlight: significant malware in 2015

Just as the marketplace grows for vulnerabilities, malware in 2015 took on a new focus. In today's environment, malware needs to produce revenue, not just be disruptive. This has led to an increase in ATM-related malware, banking Trojans, and ransomware.

ATM malware prevalence and trends

2015 bore witness to a steady increase in automated teller machine (ATM)-related attacks. While not new-ATM malware attacks have existed since 2004[268]—they have risen in frequency exponentially the past few years.[269] ATM-targeting malware, as reported by our telemetry, occurs in moderate but still alarming numbers. Due to the oscillated nature of targeted platforms, once compromised they appear on the radar of AV companies and major news feeds. These attacks are most likely only reported back to the banks and ATM manufacturers, making it difficult to gather reliable telemetry.

Attacks targeting ATMs generally fall into one of two categories:

- Stealing credit card information. These attacks may use hardware such as skimmers, software loaded onto the ATM, or a combination of both.
- Directly dispensing cash. These attacks rely on directly bypassing card authentication and are performed at the software level.

While there's no definitive answer as to what contributes to the rise of ATM malware, it is likely that an aging ATM fleet plays a significant role. The ease of access to the inner workings of certain ATMs and their locations contribute as well. What is certain is that cybercriminals attacking ATMs are well-organized and operate on an international scale.[270]

Targeted platforms

ATM software architecture follows the design of the common architecture and consists of the following major blocks:

- Hardware modules
- Service providers' drivers for hardware modules
- Financial services extension (XFS) manager
- Business application

Maintained and promoted by the European Committee for Standardization[271] (CEN), the XFS manager is based on earlier work of Windows Open Architecture Extension for Financial Services by Microsoft (WOSA/XFS). The CEN/XFS manager communicates to the service providers, which in turn implement device drivers for particular hardware modules. In doing so, CEN/XFS abstracts a business application (e.g., money dispenser, secure crypto-processor, a money vault, a pin-keyboard, a card reader, a printer) from ATM hardware modules by providing a set of standard APIs. The business application receives, interprets, and acts on user inputs ultimately rendering services exposed by the ATM. Ninety-five percent of ATMs use a locked down version of Windows XP,[272] despite the fact that Windows XP is no longer supported by Microsoft.[273] The ATMs are not expected to be connected to the Internet without an encrypted transport layer, such as a Cisco VPN tunnel.[274] Small ATMs found in kiosks, small shops, and service stations can also utilize proprietary telemetry links using ISM bands[275] or phone lines.

ATM malware prevalence

Standalone ATMs do not typically run endpoint security software or provide any telemetry to gain knowledge about the prevalence and distribution of malware. For insight into the prevalence and distribution of ATM malware, we monitored the crowd-sourced sample gateway VirusTotal.[276] Operating under the assumption that malware targeting ATMs has a high probability of traversing client systems and being spotted by users or AV engines before installing, as well as knowing that the XFS manager is implemented via msxfs.dll (in the Microsoft Windows environment) a simple YARA[277] rule was crafted to identify files linked with msxfs.dll and importing a set of XFS APIs common to known ATM malware. We limit our observations to three common AV engines—Kaspersky, Microsoft, and ESET—based on the prevalent number of malware detections and the uniqueness and consistency of chosen malware names. It is important to note that the simplified ATM malware files selection rule would not exclude legitimate XFS applications infected by viruses. As such, these were carefully examined and filtered out. Further analyzing sourced files revealed the following ATM malware landscape throughout 2015.

[269] http://newsroom.mastercard.com/2015/05/11/security-matters-the-continuous-evolution-of-atm-fraud-attacks/.

[270] http://www.bankinfosecurity.com/atm-heist-a-8178.

[271] http://www.cen.eu/Pages/default.aspx.

[272] http://money.cnn.com/2014/03/04/technology/security/atm-windows-xp/.

[273] https://www.microsoft.com/en-us/WindowsForBusiness/end-of-xp-support.

[274] http://www.cisco.com/c/dam/en/us/td/docs/solutions/Verticals/Financial_Services/Financial_Branch_Banking/financial_banking.pdf.

[275] http://www.conceptdraw.com/examples/diagram-example-atm-system.

[276] https://www.virustotal.com/.

[277] https://plusvic.github.io/yara/.

The total number of samples within a malware family reflects all cases submitted to VirusTotal over the course of the year, including non-unique submissions and representation from different geographical regions. Having a ratio of unique submissions relative to the total number of samples, coupled with the geographical information for each individual submission source gives us a sense of prevalence and distribution for a particular malware family. Of particular note, the total number also reflects the readiness of AV engines to detect the sample at the time of the submission and does not take into account the subsequent re-scans.

When comparing these AV engines, Kaspersky's provides the best results based on granularity of malware identification. Using Kaspersky's AV engine as a baseline, and layering the overall distribution and number of detected files within individual malware families, a few interesting cases emerged.

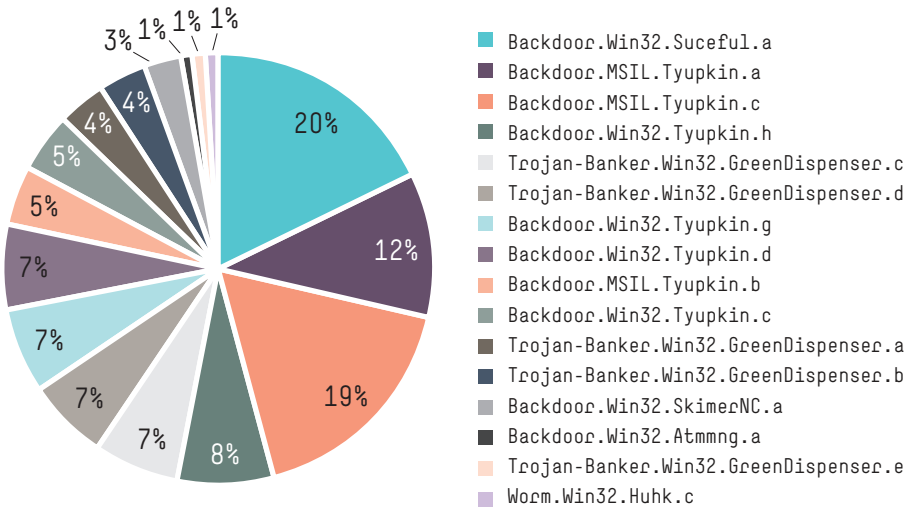


Figure 30. ATM malware rates as per the Kaspersky AV engine

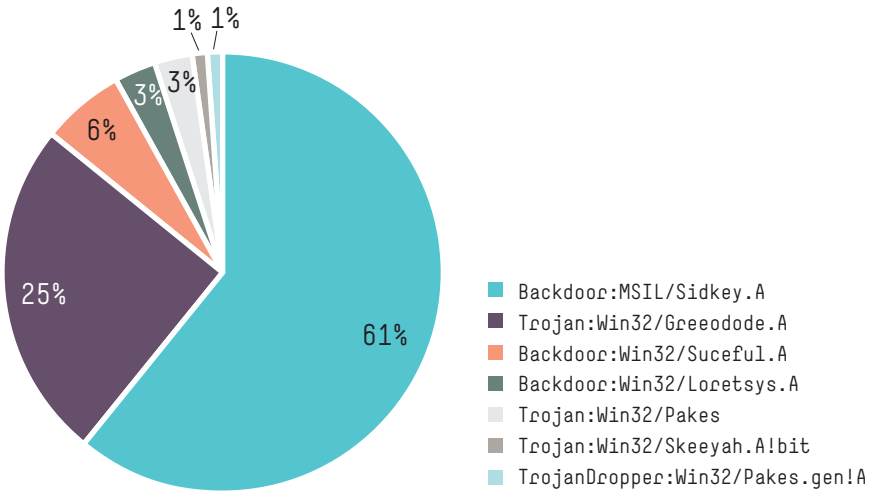


Figure 31. ATM malware rates as per the Microsoft AV engine

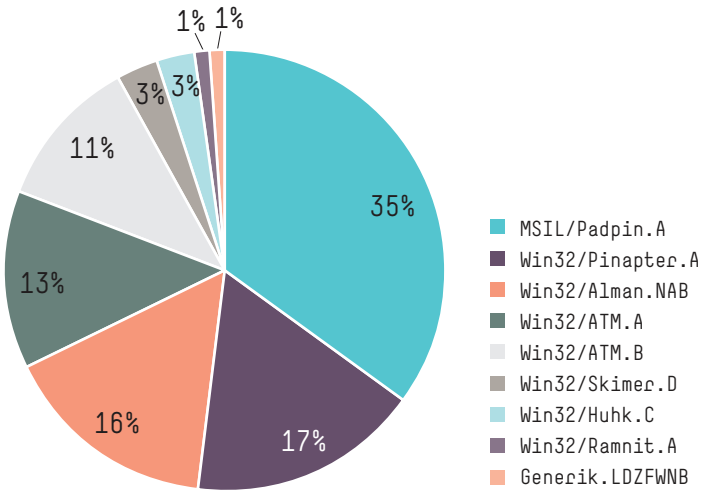


Figure 32. ATM malware rates as per ESET

AKA Suceful

Kaspersky's AV engine assigned the name Backdoor.Win32.Suceful.a to an ATM-related malware that made its debut in the middle of 2015 and so far has three distinct submissions in VirusTotal. The first file originated in Russia at the end of August (08/28/2015). Later submissions of the same file were made from countries such as Germany, India, Ukraine, Netherlands, Malaysia, Italy, South Republic of Korea, and Taiwan. The last submission occurred on September 19 from Korea.

The second file was submitted just one minute later on August 28 and originated in France. It might suggest a check by the malware actors using Internet proxies for anonymization, or it may just be a coincidence. The later submissions of this same file were made from India, Ukraine, Malaysia, Netherlands, and Taiwan. As seen in the first case, the whole distribution took about a month with the last submission made on September 23. The monthly lifespan likely indicates that most of the AV engines have caught up with the malware. It could also mean the malware finished its transitions through gateways and now resides on endpoints, such as ATMs, which are not routinely scanned.

The third file is likely a continuation of the campaign started in July. There are only two submissions made, both on September 21 and originating from Vietnam.

Such a broad range of countries of origin suggests a very wide and multinational distribution as well as possible involvement of an organized group or groups of actors.

The malware is written in Delphi.[278] The prevalent language of the file resources suggests a Russian-speaking country as a point of origin. The malware uses the WFSAsyncExecute API and also employs the WFSOpen, WFSStartup WFSExecute APIs from the msxfs.dll library. References to Pinpad, IDCardUnit, RequestID, DBD_MOTOCARDRDR, Key-ENTER, Key-CANCEL, Key-CLEAR as well as other XFS-related strings suggests ATM card reader manipulations. The malware derives its name from a typo made in an internally used return status string (SUCEFUL) during a successfully executed operation.

[278] http://www.embarcadero.com/products/delphi.

AKA Tyupkin

The name Backdoor:MSIL/Sidkey.A is used by Microsoft AV and covers most variations of the Backdoor.Win32.Tyupkin and Win32/Padpin families as identified by Kaspersky's and ESET-NOD32 AV engines respectively.

This family of malware is not new, yet it persisted throughout 2015 despite its relatively high rate of detections by AV engines. In April 2015, a file detected as Backdoor.MSIL.Tyupkin, a variant flagged by Kaspersky AV, was submitted. The first submission of the file came on April 9 and originated in Russia. Again, judging by the file's prevalent language and the initial locale of submission, it is fair to say that the malware was likely targeting Russian-speaking regions. As suggested by the detection name, this malware was written using the .NET framework and utilizes XFS framework for its malicious purposes. The most prevalent file name for this variant seen in the wild is ulssm.exe, which is hardcoded in the malware. Several notable XFS functions utilized by this sample are: WFSExecute, WFSFreeResult, WFSGetInfo, WFSIsBlocking, WFSOpen, _wfs_pin_data, _wfs_pin_func_key_detail, _wfs_pin_getdata, _wfs_result, WFSStartUp, _wfsversion, WFSCancelBlockingCall, _wfs_cdm_cu_info, _wfs_cdm_denomination, and _wfs_cdm_dispense.

The names of the APIs used suggest this malware is focused on pin-pad and money cassette dispenser manipulation. Other submissions of this variant came from the United States, Israel, Malaysia, Great Britain, France, Taiwan, Estonia, Indonesia, Czech Republic, and Brazil with the last submission made on July 18, originating in France.

The Tyupkin malware family proved to be popular among attackers. One recent and persistent submission is a Tyupkin.h variant. The first submission of this variant was made on April 15 and originated in China. The most recent was from Italy and submitted on November 4. From the dynamic of submission we observed it appears as if the different variants were released nearly simultaneously targeting different geographical regions. Some of the variants proved to be more persistent continuing to appear in the wild at the time of this writing.

As with the previously mentioned Suceful, judging by the exported XFS functions and services, Tyupkin is very pervasive and attempts to dispense cash from the ATM to the attackers. At this time, known variants of Tyupkin do not attempt to manipulate or steal the user's card.

Despite the high detection rate by AV engines, this malware family still manages to persist and due to its international prevalence and efficacy[279] is most likely used by organized crime groups mostly targeting NCR ATMs.

The names of the APIs used suggest Tyupkin is focused on pin-pad and money cassette dispenser manipulation.

[279] https://securelist.com/blog/research/66988/tyupkin-manipulating-atm-machines-with-malware/.

AKA GreenDispenser

This family of malware first came to light in the beginning of June 2015. The file is detected as Trojan-Banker.Win32.GreenDispenser.A, a variant of Win32/ATM.B, and Trojan:Win32/Greeodode.A by the Kaspersky, ESET-NOD32, and Microsoft AV engines respectively.

As well as its predecessor Tyupkin, GreenDispenser targets NCR ATMs and once installed dispenses unauthorized cash to malevolent actors.[280] It is written in C and imports the WFSGetInfo, WFSFreeResult, WFSOpen, WFSClose, WFSExecute, WFSStartUp, WFSIsBlocking APIs from the MSXFS.dll. It includes functionality to delete itself on request and simulate the ATM out-of-service message. The malware can be limited to act within a certain time frame to reduce exposure and the chance of being detected.

The initial submission to VirusTotal was made on June 4 and originated in India. Later submissions were made from the United States, France, and Japan. These submissions occurred more or less evenly throughout the following months up to September 29. The later variations of the GreenDispenser Trojan family included Trojan-Banker.Win32.GreenDispenser.b, Trojan-Banker.Win32.GreenDispenser.c, and the Trojan-Banker.Win32.GreenDispenser.d as detected by Kaspersky AV.

The Trojan-Banker.Win32.GreenDispenser.b submissions started in the second half of June, originating in Mexico, and ran up until the end of September. The submissions were made from sources in Japan, Netherlands, and France.

The Trojan-Banker.Win32.GreenDispenser.c variant submissions were only seen in September and covered regions such as France, Netherlands, and Mexico.

The final variant, Trojan-Banker.Win32.GreenDispenser.d, was initially seen in June with the first sample submitted on June 11 from a source in Mexico. Additional submissions began in August and continued until the beginning of October from countries such as Japan, France, the United States, and Russia.

[280] http://www.casefi.com/uncategorized/greendispenser-atm-malware-alert/.

From point of sale to point of steal

ATM malware prevalence is on the rise in 2015 compared to 2014. ATM malware targets financial institutions directly and adds to an already hefty portfolio of attacks on financial services such as credit card skimming devices and point-of-sale (PoS) credit card information memory scraping. The ATM attacks expose weaknesses in ATM infrastructure such as a lack of regular maintenance, misplacement of the service keys that allow easy access to the ATM software, the use of unsupported and misconfigured operating systems, and the absence of regular checks by AV solutions. Some of the attacked ATMs were located within convenience stores. Many of these stores run extended business hours and provided an easy target during evening and night hours of operation when such locations are less crowded. As seen in the prevalence and the geographical distribution data, the problem is truly global and the attacks are well organized. As a precaution, banks should constantly review the physical security[281] of ATMs in addition to updating the software that controls and protects the machines.

Banking Trojan takedowns do little to stem the scourge

In October, the banking Trojan Dridex generated a fair amount of public attention as the FBI, Department of Justice, the UK National Crime Agency, and a number of other European law enforcement and technology companies announced the arrest of an administrator and the disruption of the botnet's command and control (C&C) servers.[282] Dridex evolved from the Cridex Trojan, which itself is based on the Zbot/Zeus Trojan.[283] These newer versions better protect their communications and disseminate themselves more efficiently than their predecessors. The C&C networks of Zeus and related banking Trojans are typically encrypted and peer-to-peer capable. They utilize domain-name generator algorithms (DGAs) and a host of other anti-interception technologies to maintain the online presence required for continued operations.

[281] http://www.komonews.com/news/local/Thieves-use-bucket-loader-dump-truck-in-elaborate-ATM-theft-327457451.html.

[282] http://www.justice.gov/opa/pr/bugat-botnet-administrator-arrested-and-malware-disabled.

[283] http://blog.trendmicro.com/trendlabs-security-intelligence/banking-trojan-dridex-uses-macros-for-infection/.

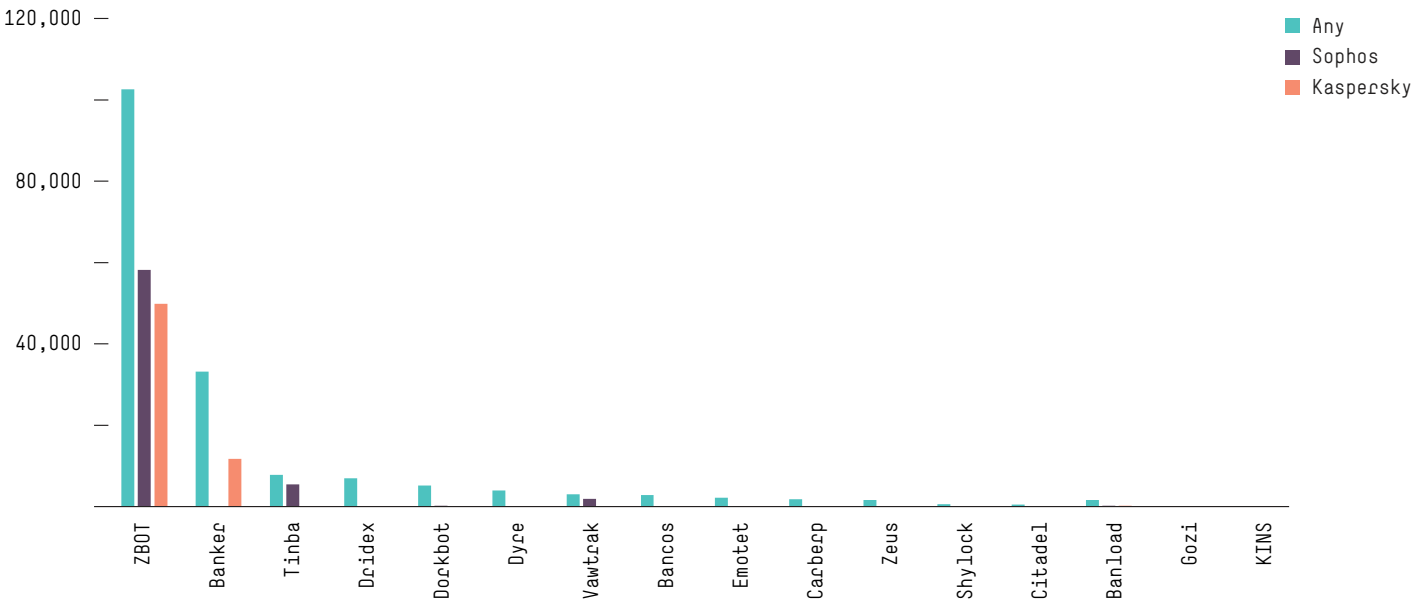


Figure 33. Banking Trojan detections in VirusTotal

Dridex infections often begin with a Microsoft Office macro chaining together multiple scripts, usually JavaScript or PowerShell, to solicit a download of the main executable. The sample W2KM_DRIDEX.YYSPE demonstrates implementation. This case is just one of many where the Visual Basic for Applications (VBA) macro invokes a base64-encoded PowerShell to construct JavaScript, which then builds x86 shellcode that solicits the download. Automated file scanning of such documents is made particularly difficult due to this blending of technologies and layering, which renders the malicious payload opaque to static analysis.

The continued success of these Trojans is likely due to the combination of two factors. The malware uses Microsoft Office documents rather than executables to disseminate itself. While many users have learned not to run programs from unknown sources, they are still likely to open documents. The Trojans also explicitly pack an executable. Anti-virus engines are more likely to identify a clearly packed file and flag it. This behavior resulted in a new approach by the malware authors. The surreptitious embedding of unpacking code in what appears to be high-level-language (HLL) compiled code, complete with WinMain, APIs, library code such as printf(), can fool even human analysts. This use of HLL compilers gives the malware binary an initial allure of legitimacy. However, it is the source-level obfuscation, realistic binary constructs, and looser emulation context that collectively contribute to resisting current generic detection strategies.

Aiding the distribution of banker malware is the resurgence of the Office Macro, which appears to be making a comeback.[284] This comes as no surprise, because finding and exploiting zero-day vulnerabilities in Office applications is not as trivial a task as compared to obfuscating a bit of VBA.

[284] http://www.theregister.co.uk/2014/07/08/macro_viruses_return_from_the_dead/.

It appears that Microsoft Office Word documents and Excel® spreadsheets remain the favored attachments. Many businesses use these programs to conduct day-to-day operations, which provides a broad user base for attackers to target.

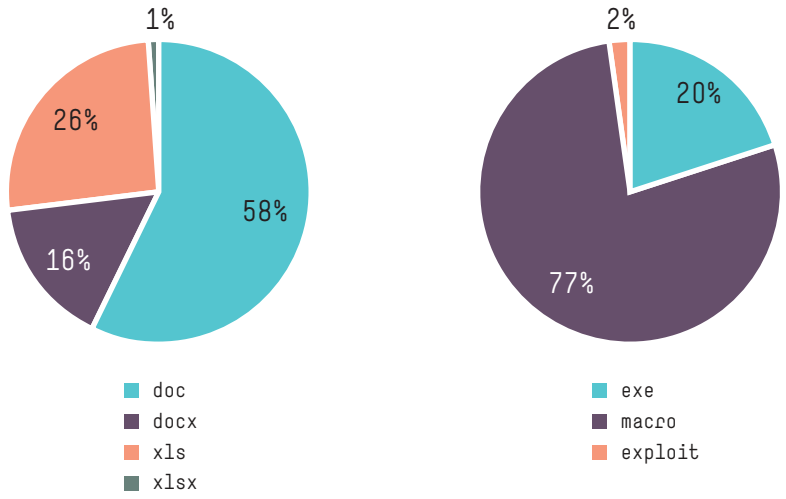


Figure 34. Office files with macros, VirusTotal 2015

Ransomware

This past year saw a number of ransomware families, including Cryptolocker,[285] Cryptowall,[286] CoinVault,[287] BitCryptor,[288] TorrentLocker,[289] TeslaCrypt,[290] and others, wreaking havoc by encrypting files of private and corporate users alike. Once encrypted, the malware author typically demands ransom in exchange for decryption keys required to restore the files. In coordinated takedowns between law enforcement and security researchers, some ransomware operations were stopped, or at least slowed. This often includes taking over the command and control infrastructure, which contains the decryption keys. One excellent example is the CoinVault takedown by Kaspersky Labs and the Netherland National High Tech Crime,[291] which exposed over 14,000 decryption keys.

The best protection against ransomware is a sound backup policy for all important files on the system. By default, Windows keeps shadow copies of the files in the user's home folder. Sometimes the system can be recovered from a ransomware attack by restoring shadow copies, but ransomware authors will try to disable shadow copy restores by deleting them.

Hiding in plain sight

As security products improve at inspecting and identifying packed or unusual code, malware authors appear to be moving toward blended scripts to prevent detection. Over the past year, there has been a general shift from wholesale packing of malware executables toward better utilizing the inherent strengths of high-level language compiled binaries (i.e., HLL/C, MFC, .NET, Visual Basic, etc.) and off-the-shelf scripting engines (i.e., AutoIt, cscript, vbscript, PowerShell, etc.). Regardless of the type of malware, this combination provides a new level of difficulty in detection and eradication.

Binaries compiled in .NET, Visual Basic, and MFC are less trivial to emulate and traverse. This allows for malicious functionality to be more easily hidden, from both inexperienced malware analysts and automated scanning tools.

[285] http://blog.trendmicro.com/trendlabs-security-intelligence/crypto-ransomware-sightings-and-trends-for-1q-2015/.

[286] http://www.computerworld.com/article/3012101/security/pony-angler-and-cryptowall-mixed-into-dangerous-cyberthreat-cocktail.html.

[287] https://securelist.com/blog/virus-watch/67699/a-nightmare-on-malware-street/.

[288] http://www.theregister.co.uk/2015/11/02/kaspersky_announces_death_of_coinvault_bitcryptor_ransomware/.

[289] http://www.scmagazineuk.com/torrentlocker-copycat-cryptofortress-leads-new-wave-of-ransomware/article/402279/.

[290] https://securelist.com/blog/research/71371/teslacrypt-2-0-disguised-as-cryptowall/.

[291] https://noransom.kaspersky.com/.

Outlook for 2016

Increasingly skillful cybercriminals seem highly motivated and intent on stealing our identity, emptying our bank accounts, and holding our data for ransom. This threat is compounded by the ever increasing complexity of running a modern IT enterprise. For businesses and their IT departments, vigilance needs to be fortified with action and preparedness.

End users and security professionals alone should not be shouldering the burden of ensuring the storage and integrity management of data. Security product vendors must become as nimble as the adversary by staying abreast of the multitude of emerging techniques—even while constrained by existing business factors. Programming errors, system oversights, ill-conceived features, and poor QA are not assisted by time-to-market and resourcing pressure justifications. The adversary is determined. The defense must be more so.

	HLL/C	MFC	.NET	Visual Basic
2011	1	•	115	•
2012	377	•	399	•
2013	652	23	863	•
2014	560	167	2947	28,054
2015	20.348	227	171,750	139,654

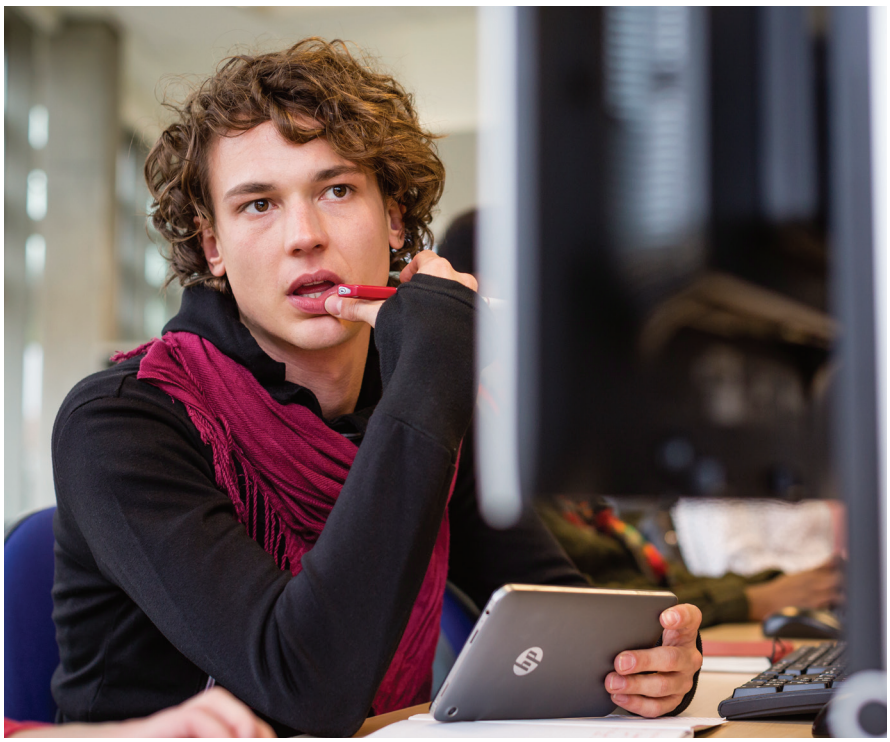
Figure 35. Languages used in malware creation 2011-2015

Conclusion

While the apparent stagnation in the overall growth of malware is an unexpected positive, the slow shift of focus away from Windows toward Linux, Android, and OS X means the overall attack surface for malware continues to grow. While always disruptive, today's malware has become focused more on money than disrupting services. For these non-Windows platforms, malware often takes the shape of potentially unwanted applications, which could confuse a non-technical user as to what is or isn't malware. This is especially troubling given the first signs that Apple's walled-garden application store approach may not be infallible. While the anticipated flood of attacks on Internet of Things (IoT) devices has yet to occur, attacks on home routers[292] may be a precursor of things to come.

The ever-present ATM has become the focus for many types of attacks, with malware authors targeting the users of ATMs and the machines themselves. While there have been coordinated law enforcement takedowns of banking Trojan infrastructure, statistics show the attackers are capable of restoring services to the botnets in a surprisingly rapid fashion. As more and more of our financial transactions occur online, criminals will continue to target these transactions for profit. Put simply, if there is money to be made, there is money to be stolen. The industry must focus on securing these transactions to deprive attackers of the illicit income they so desire.

[292] http://www.computerworld.com/article/2921559/malware-vulnerabilities/malware-infected-home-routers-used-to-launch-ddos-attacks.html.



Software analysis

In order to have a consistent view of the data analyzed for this report, all identified issues were classified according to the HPE Software Security Taxonomy (originally the "Seven Pernicious Kingdoms"[293]), which was substantially updated and refined[294] in mid-2014.

During 2015, the taxonomy extended further to include other assessment techniques (such as manual analysis and mobile vulnerabilities) and HPE Security Fortify products such as HPE Security Fortify on Demand (FOD). This endeavor continues as the taxonomy evolves with the most recent updates considered for analysis in this report. Changes to the taxonomy that affect statistics presented in this report are flagged as necessary throughout the text. As a reminder of how the taxonomy works, findings are sorted into kingdoms, which consist of vulnerability categories, each of which includes one or more security issues.

[293] https://cwe.mitre.org/documents/sources/SevenPerniciousKingdoms.pdf.

[294] http://community.hpe.com/hpeb/attachments/hpeb/off-by-on-software-security-blog/402/1/An%20Evolving%20Taxonomy%20-%202014.pdf.

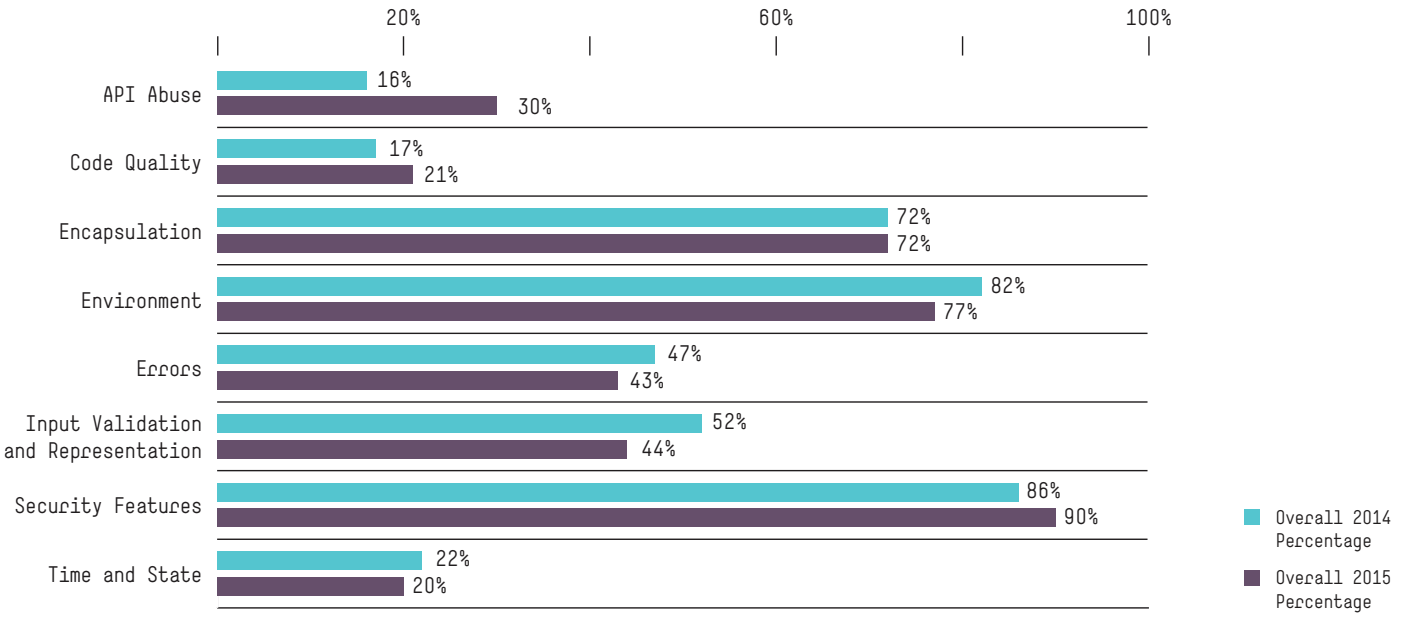


Figure 36. The likelihood per kingdom of apps found to be vulnerable

Why and how we do this analysis

Our yearly analysis of trends in application security provides a unique snapshot of the state of application security during the past year. Readers are encouraged to use our findings as a guide for issues to watch for during their own development processes, to help plan their own effective security development lifecycle, and to structure and deliver effective security training.

Before diving into the data, let's define it. First, the dataset is separated into two groups: mobile applications and those not geared toward mobile (in this Risk Report, referred to simply as "mobile" and "apps" or "web apps"). The issues affecting the two types continued, in 2015, to differ significantly. Separating the datasets provides a clearer picture of which vulnerabilities occur in each group. It also accounts for significant differences in sample size, which are predicated on significant differences in dataset size. Each of the two main datasets is drawn from the applications processed by the HPE Security Fortify on Demand service between October 30, 2014, and October 30, 2015. This data was anonymized and sanitized. The apps dataset, drawn from over 7000 scanned applications, includes both web and desktop apps but, as previously noted, excludes mobile. The mobile dataset is drawn from over 450 scanned apps. Note that though both datasets have expanded in size since last year's Risk Report, the rate of increase for mobile is double that of apps—a 20% increase as opposed to apps' 9% bump.

Application results

Figure 36 shows the percentage of applications that exhibited a vulnerability in the given kingdom at least once, but provides no information on what percentage of applications had more than one vulnerability there.

Generally, the breakdown between the two years is similar. Year-to-year changes in the rankings for the three kingdoms with the lowest representation (API Abuse, Code Quality, and Time and State) are primarily due to changes to the HPE Software Security Taxonomy itself. The most prevalent vulnerabilities remain the same for both years. Likewise, the increase in API Abuse issues may be attributed to changes in the taxonomy. In order to make a fair comparison, disregarding the changes and comparing the values based on the older classification, API Abuse vulnerabilities were actually reduced by half from 2014, to about 8% of vulnerabilities noted.

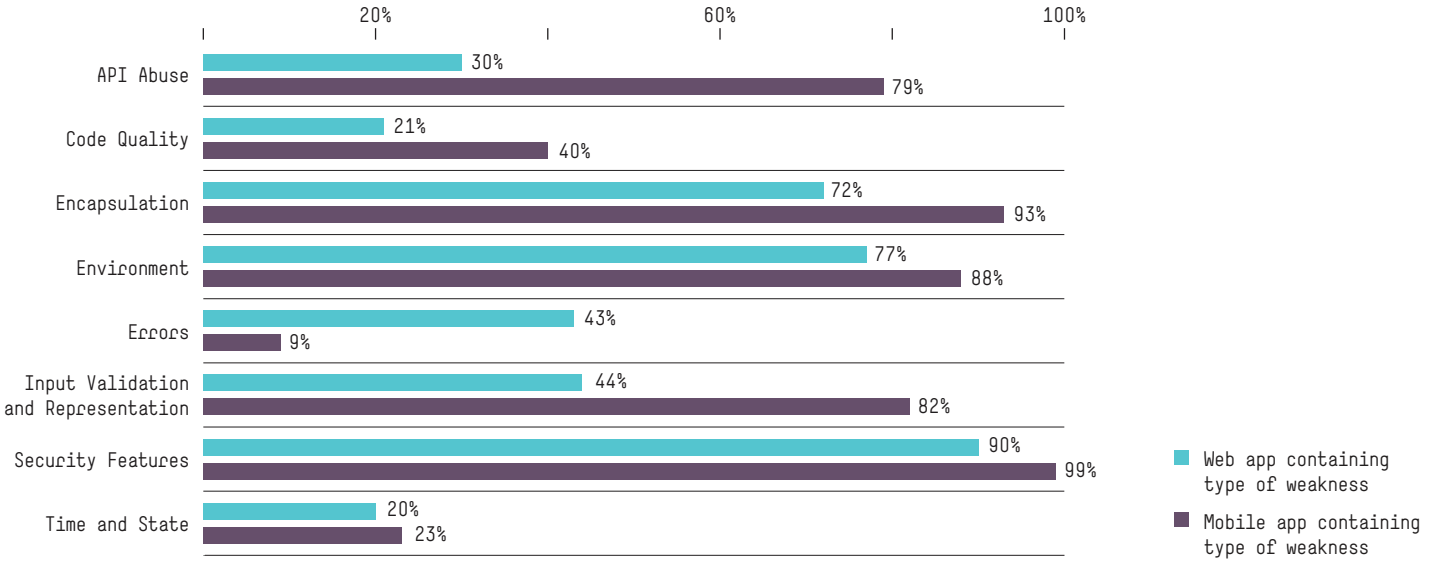


Figure 37. Comparing kingdom incidence in mobile and non-mobile applications scanned

Mobile results

Turning now to the sphere of mobile applications, which differs from the larger dataset in significant ways.

Mobile applications present different issues from those seen in non-mobile applications. Figure 37 compares the likelihood that mobile apps and non-mobile apps will exhibit at least one issue in each kingdom. As with Figure 36, the chart shows the percentage of applications that exhibited a vulnerability in the given kingdom at least once, but provides no information on what percentage of applications had more than one vulnerability there. (This chart includes results from HPE Security Fortify on Demand's Manual Mobile Analysis tool.)

Security Features continues to be the most represented kingdom for both traditional and mobile applications. Mobile applications tend to see over 10% more issues related to security features than do other applications. The vast majority of API Abuse issues in mobile are from three categories: Push Notifications, Ad/Analytics Frameworks, and General Pasteboard. As noted before, changes to the taxonomy affected the API Abuse kingdom. Likewise, taxonomy changes affected mobile's results in the Environment kingdom (showing a 14% apparent increase in vulnerabilities from before the changes) and in Time and State (showing a 10% decrease).

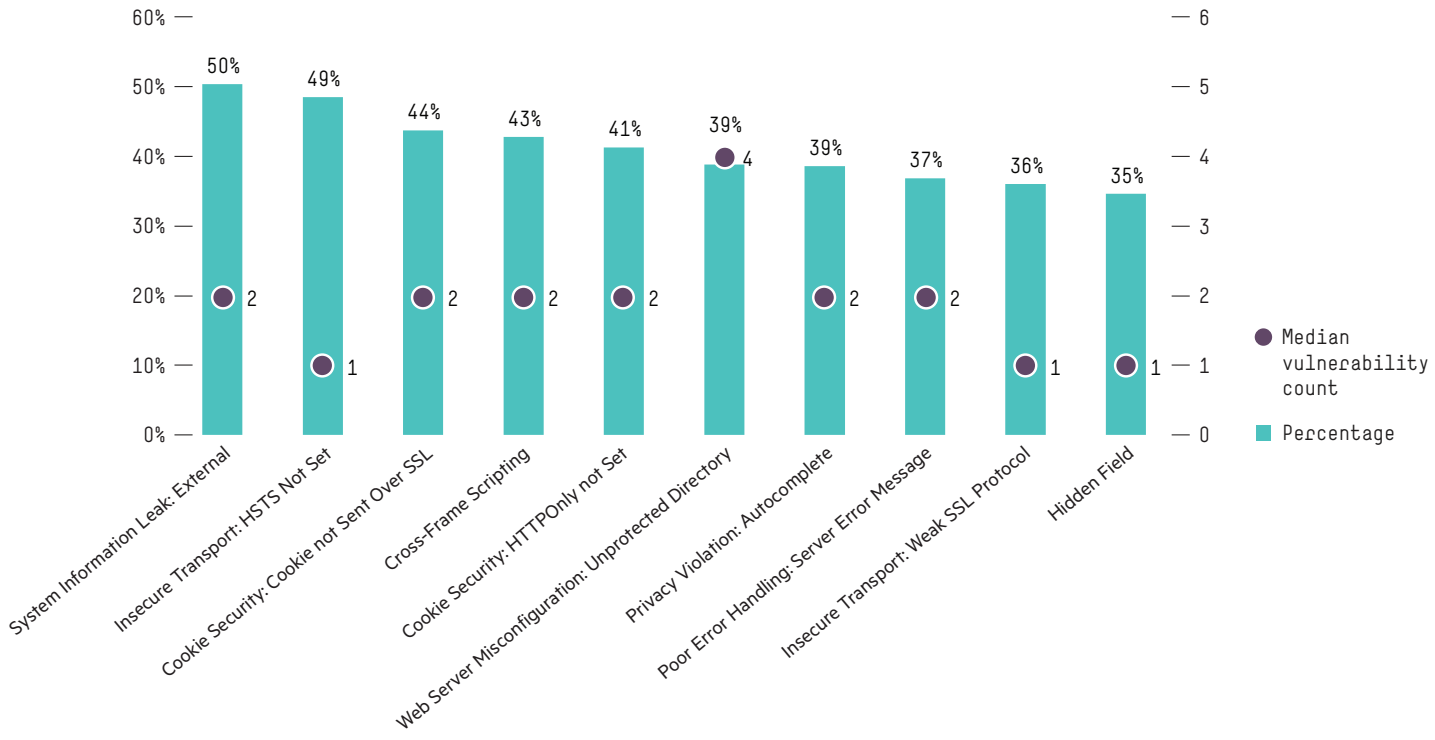


Figure 38. The 10 most commonly occurring vulnerabilities in the applications dataset

Top vulnerablities in applications

The chart in Figure 38 shows the 10 most commonly spotted vulnerabilities in non-mobile applications in 2015. Four of the kingdoms—Encapsulation, Security Features, Environment, and Errors—are represented in this chart.

Once again, a prominent result looks more remarkable than it is. The System Information Leak: External category, which didn't make an appearance in last year's analysis, would seem to be at the top of the heap in 2015. The numbers aren't actually as overwhelming as the bars would indicate—another artifact of changes to the taxonomy—but the vulnerability itself is well worth examining. It describes a situation in which too-detailed error messages leak system data that might help attackers gain dangerous visibility into the system. It's not in itself a critical-severity vulnerability, but it can certainly assist in other attacks, including critical ones. It is troubling to note in the analysis that half of the applications scanned exhibited a median of two System Information Leak: External issues per application. The Insecure Transport: HSTS Not Set category, second from the left in Figure 38, is a fairly new taxonomy entry. The HTTP Shared Transport Security (HSTS) header was introduced to battle man-in-the-middle (MitM) attacks over SSL/TLS, such as protocol downgrade and cookiejacking. The good news is that most modern browsers now support the HSTS header. The bad news is that the analysis indicates many applications do not yet take advantage of it.

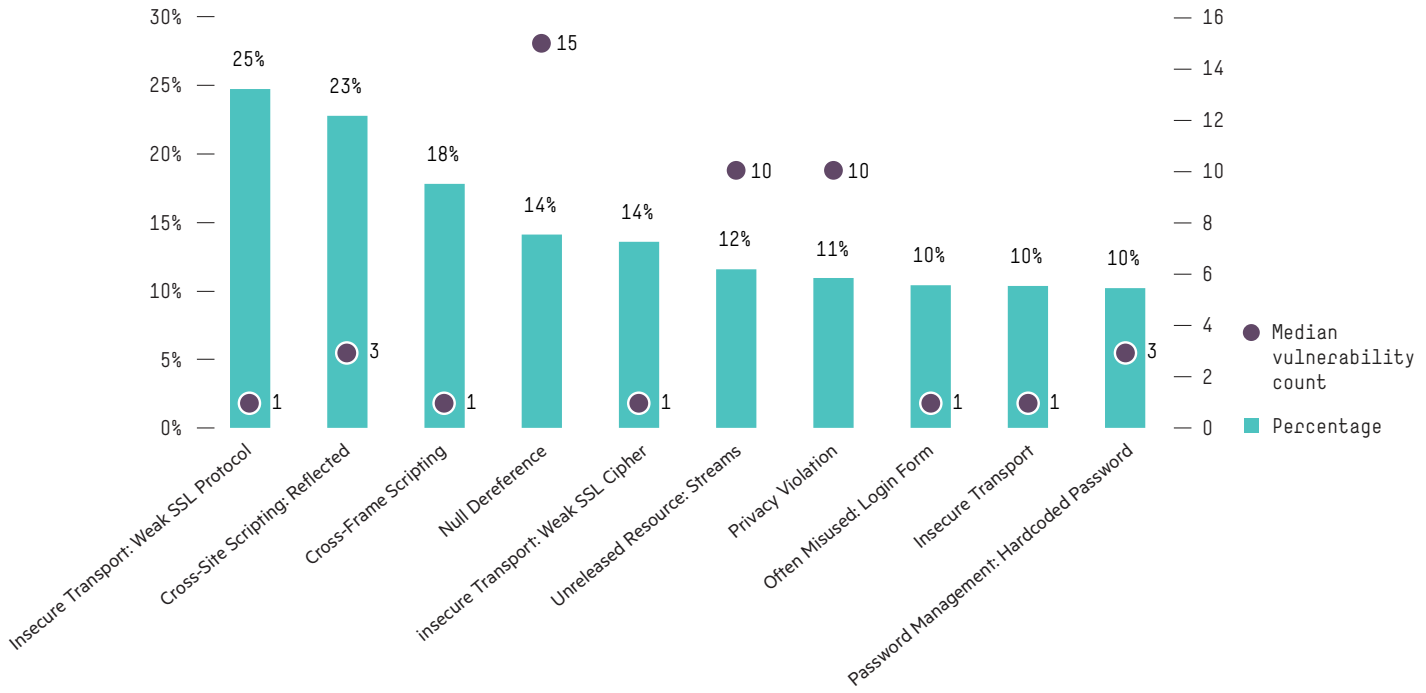


Figure 39. The 10 most commonly occurring critical-class vulnerabilities in the applications dataset

Of course, mere prevalence isn't what makes a vulnerability vicious. Figure 39 pares down the dataset to show the 10 most frequently spotted critical-severity vulnerabilities in applications. Here, the most strongly represented kingdoms are Security Features, Input Validation and Representation, Encapsulation, Code Quality, and API Abuse. Environment and Errors issues for the most part do not rise to the critical level.

Over one-third of the applications scanned—35%—exhibited at least one critical- or high-severity vulnerability.

Two categories, Insecure Transport: Weak SSL Protocol and Cross-Frame Scripting, appear in both the common and the critical top 10. Indeed, most issues in those categories are sufficiently dangerous to merit the most severe classifications. The SSL finding is worth looking at more closely. As longtime observers of the vulnerability scene know, SSL-related problems made a very big splash in the last months of 2014, with the POODLE exploit extending its reach from SSLv3 (CVE-2014-3566) to TLSv1 (CVE-2014-8730).[295] Statistics for these vulnerabilities are available for the first full year in 2015, and it's disheartening to see them make such a strong showing. It's likely that many applications continue to use weak SSL protocols and ciphers for backward-compatibility purposes, but it's still a dangerous choice.

The presence of Privacy Violations on the list and the high occurrence median—10—are daunting in their own way. Elsewhere, the critical-severity chart delivers a head-slap moment with hardcoded passwords appearing in no less than 10% of applications. As if that wasn't bad enough, the median number of occurrences in affected applications was three. Five years after Stuxnet made clear the profound security shortcomings of making a "password" part of the code itself, this particular vulnerability should embarrass any software architect that allows it to happen.

Over one-third of the applications scanned—35%—exhibited at least one critical- or high-severity vulnerability.

[295] https://www.globalsign.com/en/blog/poodle-vulnerability-expands-beyond-sslv3-to-tls/.

Figure 40. The five most frequently spotted categories across applications

Top five vulnerability categories in applications

Back to the apps dataset. Figure 40 shows the five vulnerability categories most likely to appear in an application. They come to us from the Security Features, Environment, and Encapsulation kingdoms.

Let's discuss these five more closely to determine precisely what issues are turning up in each category. Insecure Transport, Cookie Security, and Privacy Violation all fall under the Security Features kingdom. While these three categories refer to the cause of a specific vulnerability in an application, the ultimate effect of issues in these categories would lead to some form of privacy violation.

Insecure Transport: HSTS Not Set is the most prevalent issue within this category, accounting for nearly 29% of findings. Again, this may be due to the fact that HSTS is a fairly new browser capability. We'll be monitoring this issue with interest in next year's Risk Report. Likewise, Weak SSL Protocol (20%) and Weak SSL Cipher (95%) both may appear as a result of relatively new industry standards and regulations (e.g., RC4 being marked as unsafe,[296] the January 2015 release of PCI SSC's Data Security Standard 3.1[297]), which result in more issues being flagged for existing configurations. Again, backward compatibility decisions may be an issue here. Missing Perfect Forward Security (6%) is another relatively recent mitigation technique making a significant showing.[298]

In the Web Server Misconfiguration category, two issues—Unprotected Directories and Unprotected File—together composed just under 64% of our findings. The Insecure Content-Type Setting header seems to contribute to a lot of misconfiguration problems, as well. Such issues are often an enabler for other problems, such as cross-site scripting. Our data indicates that around 29%of applications in the sample set are vulnerable to misconfiguration issues such as these.

With all of these insecure-transport issues turning up, we were interested to see that in many cases cookies sent over SSL aren't very secure either. Nearly 38% of the applications we saw with cookie-related issues failed to send them over SSL—an excellent example of developer misuse of SSL/TLS. Meanwhile, HTTPOnly Not Set is strongly represented, contributing to a third of cookie security issues. The HTTPOnly flag has been around for quite some time, and yet it's still missing in 41% of applications scanned.

With System Information Leak issues, it's interesting to note that over 97% of the applications that leak information do so to some external entity. Also, based on the occurrences of the various issues, System Information Leak: External contributes to more than 80% of vulnerable instances.

[296] https://blog.mozilla.org/security/2015/09/11/deprecating-the-rc4-cipher/.

[297] https://www.pcicomplianceguide.org/pci-ssc-data-security-standard-3-1-guidelines/.

[298] https://www.eff.org/deeplinks/2014/04/why-web-needs-perfect-forward-secrecy.

Finally, let's look at privacy violations. While autocomplete issues occur in well over a third of the applications (38%), generic privacy-violation issues corresponding to the exposure of user data seem to occur more often within the affected apps. In addition, most autocorrect issues are of relatively low severity, while other frequently seen privacy-violation issues are of critical or high severity (as we saw in Figure 39). We also noticed that while autocomplete issues occurred a median of twice per affected application, critical-severity privacy issues occurred a median of 10 times per affected application.

Mobile vulnerabilities

Returning to the mobile sphere to see what issues most plagued scanned apps there, see Figure 41.

For mobile applications, it's internal system information leaks that lead the most common list. Their very large presence atop the list of most frequently encountered mobile vulnerabilities indicates that a substantial majority of the applications we saw are storing sensitive information on devices that can be left on restaurant tables, stolen from backpacks, and dropped in toilets. Note the high showing of Weak Encryption (69%), a flaw one sees far less often in non-mobile applications.

As before, the issues that are common are not necessarily critical-level. Insecure Transport, eighth on the list of common flaws, leads the list of critical issues. A quick skim of the category names represented confirms that developers are still struggling with privacy issues on these devices. Overall, it should be noted that 75% of the mobile applications scanned have at least one critical- or high-level finding, compared to 35% of non-mobile applications.

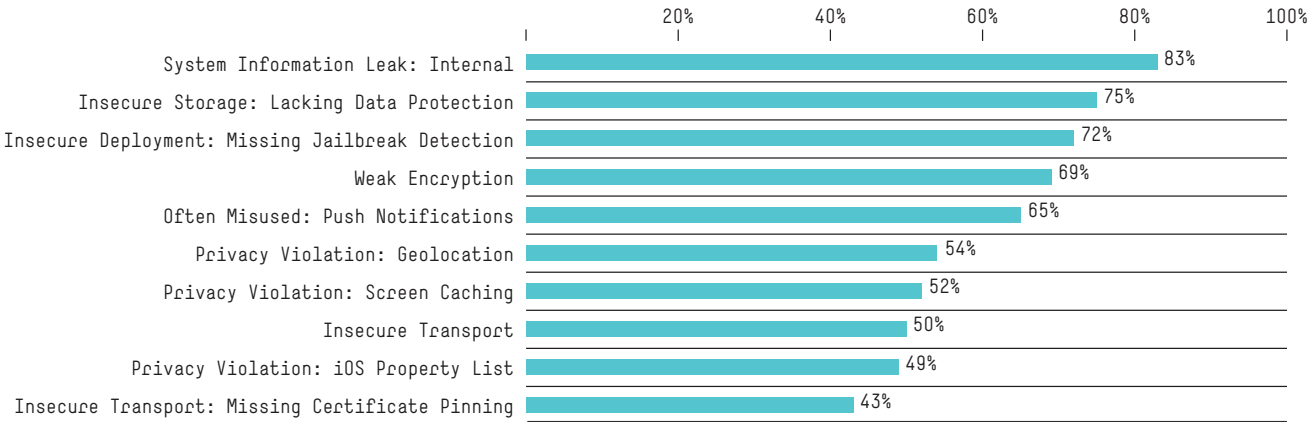


Figure 41. The 10 most commonly occurring vulnerabilities in the mobile applications dataset

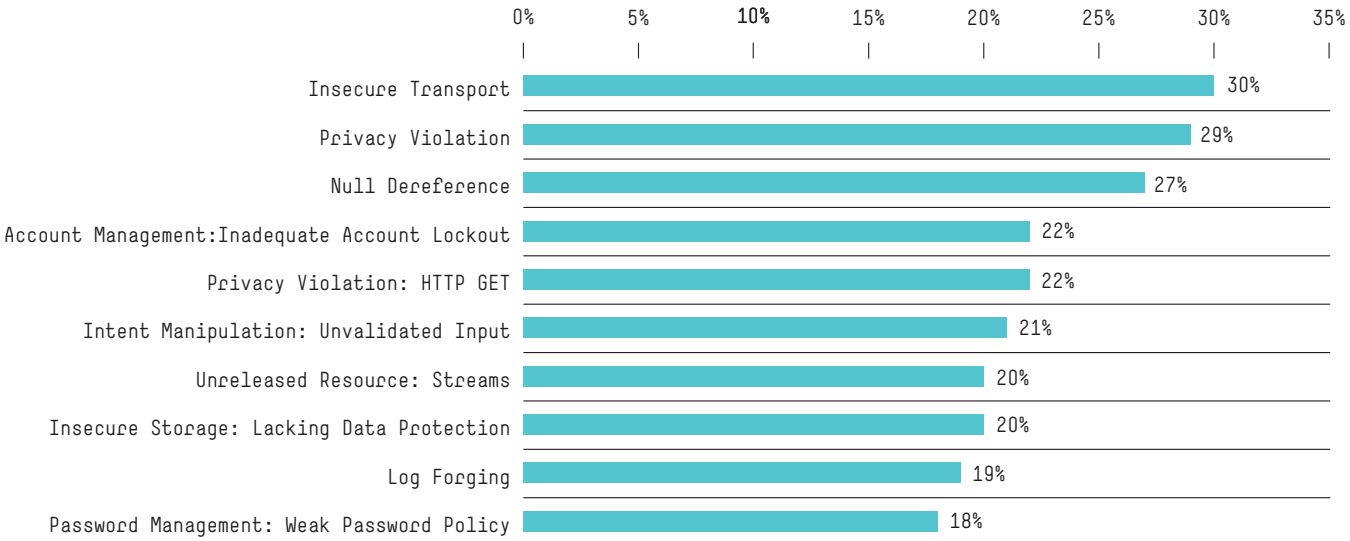


Figure 42. The 10 most commonly occurring critical-severity vulnerabilities in the mobile dataset

Figure 43. The five most frequently spotted mobile vulnerabilities

Top five vulnerability categories in mobile

The reason for that gap is clear—and it's not only a question of the size and portability of the device. Mobile applications have a different runtime environment than do traditional desktop and enterprise server applications. On mobile, the environment makes heavy use of unique personally identifiable information, such as geolocation and screen/keyboard caching. Moreover, the environment often contains both trusted and untrusted applications, creating a unique situation in which the storage and transmission of private and sensitive information becomes a more pressing issue. As such, it isn't surprising that developers appear to be introducing an equally large percentage of Insecure Storage and Privacy Violation weaknesses into their mobile applications (88%). Compare this to the top five vulnerabilities in the applications space (Figure 40), where Privacy Violations, though still making the list, have a relatively weak showing at 48%.

Within the Privacy Violations category for mobile, Geolocation, Screen Caching, iOS Property List (iPhone only), and HTTP GET issues occur at very nearly the same frequency and together account for 85% of all findings. Less frequently seen issues include problems related to credit cards, passwords, and autocomplete handling.

In the Insecure Storage category, 58% of issues spotted involve a lack of data protection. Two issues, Android Backup Storage and Android World Readable or Writeable, are only found on that platform and together account for 28% of findings.

In the category of System Information Leak weaknesses, 86% of the mobile applications scanned had at least one detected issue, while 52% of traditional applications had sensitive system information leaks detected. While the majority of System Information Leak issues, 71%, were detected as local to the device, another 24 percent had the potential to leak sensitive information outside of the device.

Often Misused comes in at the fourth spot (78%) for mobile apps as seen in Figure 43; in the sphere of traditional apps, it shows up at an anemic 13th on the frequency list, with just 23% of those applications having trouble of this sort. It's not entirely clear why this might be the case. Perhaps the problem is that mobile development is less mature than older application types, and as such, developers may not be following mature best practices for mobile-specific APIs and frameworks (e.g., Push Notifications, Ad/Analytics Frameworks, General Pasteboard, Shared Keychain, and Calendar).

Rounding out the top five most frequently seen categories of mobile flaws, the Environmental category of Insecure Deployment takes the fifth spot, with 75% of mobile applications exhibiting weaknesses such as Missing Jailbreak Detection, Malicious Behavior, and OpenSSL issues. It is interesting to note that, had we expanded Figure 43 to show the 10 most frequently seen categories of vulnerabilities, positions six through eight on that list are held by categories related to Security Features (in addition to Privacy Violation)—specifically, Weak Encryption (70%), Insecure Transport (67%), and Privilege Management (49%).

Vulnerabilities in open source software

Just as we examined trends in commercial software as seen in the customer applications submitted to the Fortify on Demand service, we examined similar trends in open source software. We used HPE Security Fortify Open Review (FOR) data on 287 applications spanning more than 10 programming languages. The dataset used for our analysis did not include any mobile software.

The two most prevalent languages in the FOR dataset are PHP (50% of all applications) and Java (28% of all applications), which is reflective of the state of open source software today—with the possible exception of JavaScript, which is gaining in popularity and ubiquity. Because the dataset for other languages is statistically insignificant, our analysis focuses on PHP and Java applications.

It's important to analyze open source applications separately from libraries and frameworks. The Fortify Open Review uses the HPE Security Fortify Static Code Analyzer (SCA) to perform security analysis. When analyzing an application, the static analyzer has a complete picture of all the dataflow traces and execution paths for that application, and therefore can provide end-to-end dataflow analysis results. Analyzing libraries and frameworks separately from applications built on top of them is different, because libraries and frameworks only provide some intermediate steps of the application's flow and therefore will not generate the same full execution paths and data flows as those of an end-to-end application. Figure 45 demonstrates the breakdown of Java and PHP applications by type. For the purposes of our analysis, libraries and frameworks are treated the same.

Language	Number of applications
PHP	143
Java	80
Python	19
.NET	14
CFML	13
Ruby	5
Other	5
MBS (Fortify SCA intermediate language)	4
SQL	2
C/C++	1
XML	1
Total	**287**

Figure 44. A breakdown of the FOR dataset by programming language

Type	PHP	Java
Application	78	33
Framework	52	27
Library	13	20
Total	**143**	**80**

Figure 45. A breakdown of analyzed PHP and Java applications by type

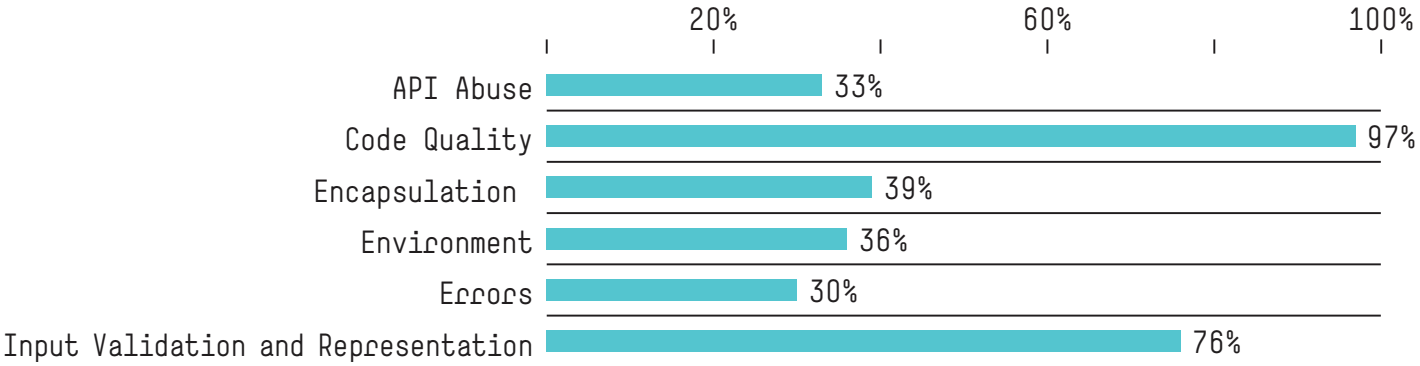


Figure 46. HPE Security Fortify taxonomy kingdom distribution across all Java applications in the FOR dataset

Distribution by kingdom: applications

Let's start by looking at the distribution of taxonomy kingdoms across applications in our dataset.

Figure 46 shows the percentage by kingdom of open source Java applications that have at least one vulnerability, and presents the distribution across all of the Java applications in the FOR dataset. Figure 47 illustrates the same data, but for PHP applications.

It is interesting to observe that in the case of Java, Code Quality is an obvious "winner," with 97% of applications vulnerable to issues from this kingdom. We have seen quite a few Code Quality findings in Java applications, including unreleased resources such as sockets and databases, and dereferences of null values. On the other hand, the HPE Security Fortify Static Code Analyzer (SCA) does not detect code quality issues in PHP out of the box, which explains why no PHP applications contain any issues from the Code Quality kingdom.

Instead, the most active kingdom for PHP is Input Validation and Representation (97%). This makes a lot of sense when one considers notoriously numerous cross-site scripting findings and CVEs in open source PHP applications. Input Validation and Representation takes third place across Java applications, with 76% of applications vulnerable to issues in this kingdom. In our experience, there are a lot more libraries for doing input validation for Java than there are for PHP, which we believe explains why the percentage across Java applications is lower than that across PHP applications. Security Features takes second place for both Java (82%) and PHP (87%) applications. Both types of applications contain a number of password management and privacy violation issues that belong to the Security Features kingdom. Their presence implies that these applications do not take good care of private data. Fortify Static Code Analyzer does not detect Time and State and Errors issues in PHP out of the box, which is why the percentage of PHP applications that contain these kinds of findings is zero percent for both kingdoms. On the contrary, 64% of Java applications contain issues related to Time and State. Specifically, a number of Java applications incorrectly use double-checked locking patterns.

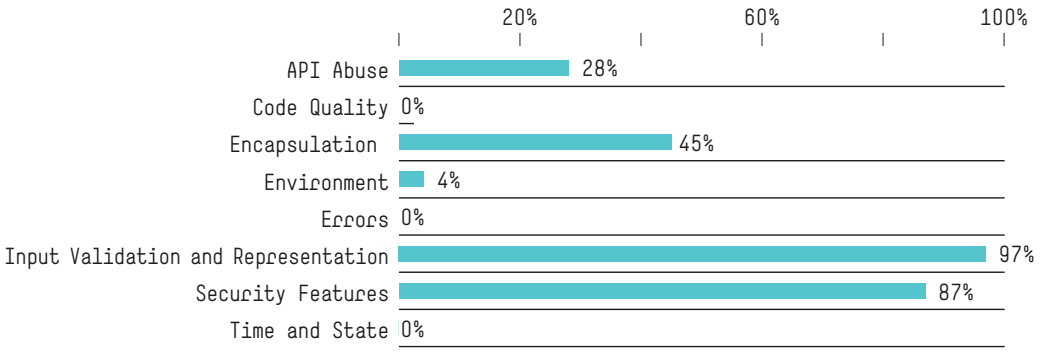


Figure 47. HPE Security Fortify taxonomy kingdom distribution across all PHP applications in the FOR dataset

Comparing the trends in open source software to those in commercial software, there are a number of interesting observations. First of all, the Security Features kingdom is prominent, with over 80% of both open source and commercial applications vulnerable to issues in this kingdom. This implies that both types of applications have trouble managing private data. The reason a much higher percentage (77%) of commercial applications are vulnerable to issues in the Environment kingdom, as opposed to open source Java (36%) and PHP (4%) applications, is because in addition to static analysis, commercial applications also underwent dynamic analysis, which is much more suitable for detecting issues in the environment rather than in actual application source code. A lot fewer commercial applications (44%, as opposed to 76% of open source Java and 97% of open source PHP) are vulnerable to issues in the Input Validation and Representation kingdom. This kingdom contains vulnerabilities that have been in existence for a long time and have been tackled by security teams in our customers' organizations for a while, through either thorough code reviews or enforcement of the usage of open source and proprietary validation libraries. Similarly, a lot fewer commercial applications (21%) are susceptible to Code Quality issues than are open source Java applications (97%). This seems to indicate that developers of commercial applications do a much better job of releasing resources and doing null checks before dereferencing a value. This could be due to their access to better tools for finding memory management issues during the testing cycle.

The only other outlier seems to be the Encapsulation kingdom, where more commercial applications (over 70%) but only 39% of open source Java and 45% of open source PHP applications contain issues in this kingdom. Most of the reported issues have to do with leaking system information outside of the application. Because commercial applications were analyzed both statically and dynamically, and system information leaks can be found both statically and dynamically, more commercial applications exhibited issues in the Encapsulation kingdom during our scans, as compared to open source applications.

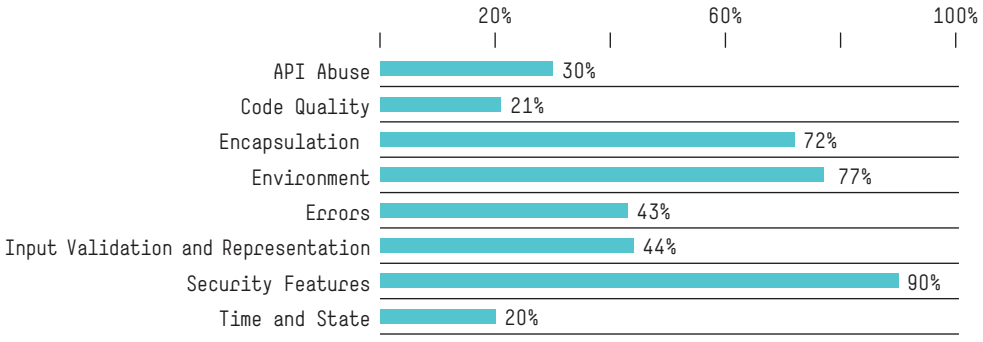


Figure 48. HPE Security Fortify taxonomy kingdoms distribution across all applications in the FOD dataset

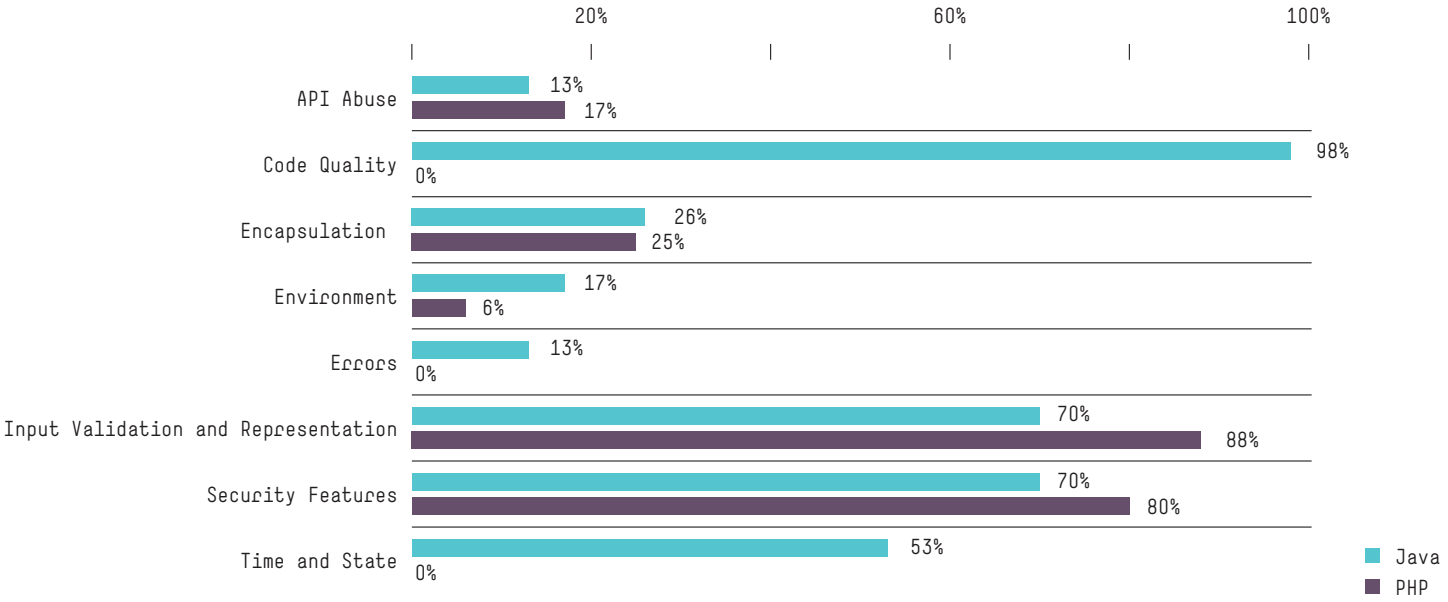


Figure 49. HPE Security Fortify taxonomy kingdom distribution across all Java and PHP libraries and frameworks in the FOR dataset

Distribution by kingdom: libraries

Now let's take a look at the same trends in open source libraries and frameworks.

Figure 49 displays kingdom distribution across all Java and PHP libraries and frameworks in the FOR dataset. The results look pretty similar to those observed for open source applications. The Code Quality kingdom is the most prolific for Java libraries and frameworks, with almost identical incidence percentages (97% for applications and 98% for libraries). If the 1% difference has any significance, it may be that libraries are exposing APIs for opening and closing resources, which would mean that managing them is left to the application. The number of libraries that contain Input Validation and Representation vulnerabilities, while still high (70% for Java and 88% for PHP), is a little lower than the number of applications (76% for Java and 97% for PHP) because, as explained earlier, libraries and frameworks represent only a step in a flow of data through the application built on top of these libraries and frameworks.

The same is true for Security Features. The number of libraries that contain issues in this kingdom (70% for Java and 80% for PHP) is slightly lower than the number of affected applications (82% for Java and 87% for PHP) because two major categories from this kingdom represented in the dataset—Privacy Violation and Password Management—usually involve flow of data, which cannot be provided end to end in a library as opposed to the application built using it. Similar conclusions can be made about the Encapsulation kingdom represented by System Information Leak issues—another dataflow category. As for Environment, more Java applications (36%) than libraries (17%) contain issues in this kingdom because it's usually the application that needs to be configured to use a particular framework, not the framework itself. The situation is different for PHP: The number of applications (4%) that contain issues in this kingdom is about the same as the number of libraries (6%) because fewer misconfiguration patterns specific to certain PHP frameworks are supported by the Fortify SCA out of the box.

Open source vulnerabilities

Next, let's look at the top 10 vulnerability categories across applications and libraries.

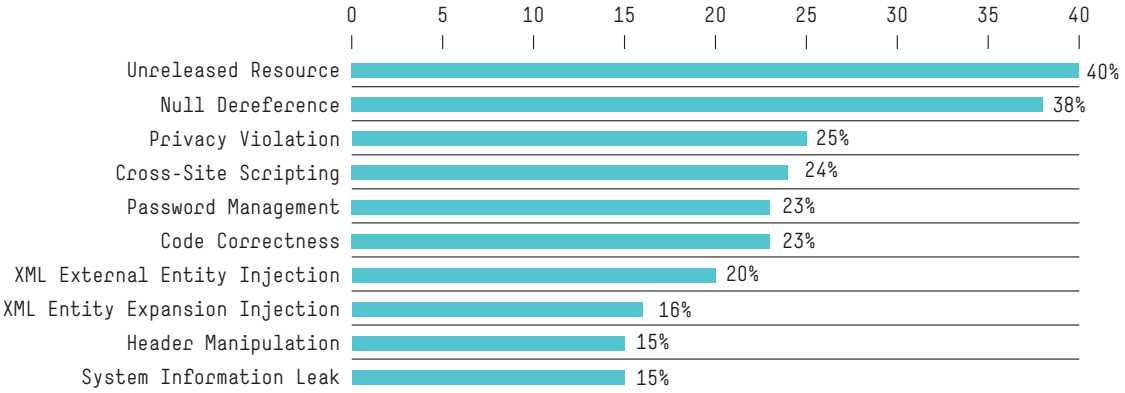


Figure 50. Top 10 vulnerability categories for Java applications in the FOR dataset

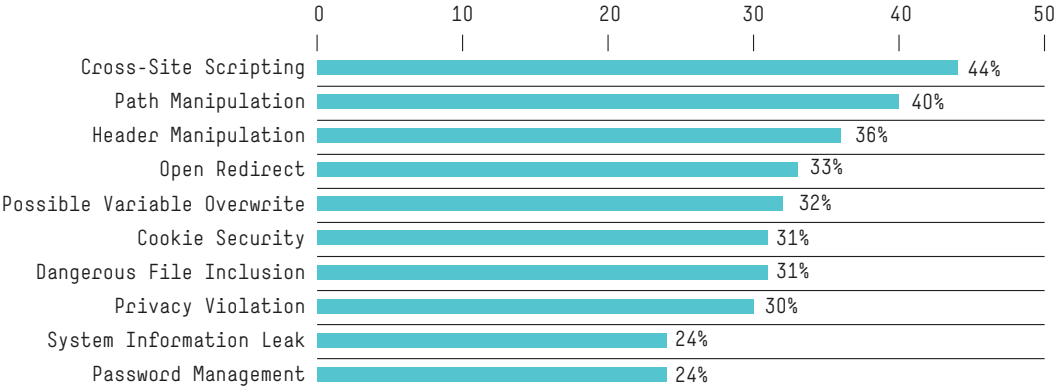


Figure 51. Top 10 vulnerability categories for PHP applications in the FOR dataset

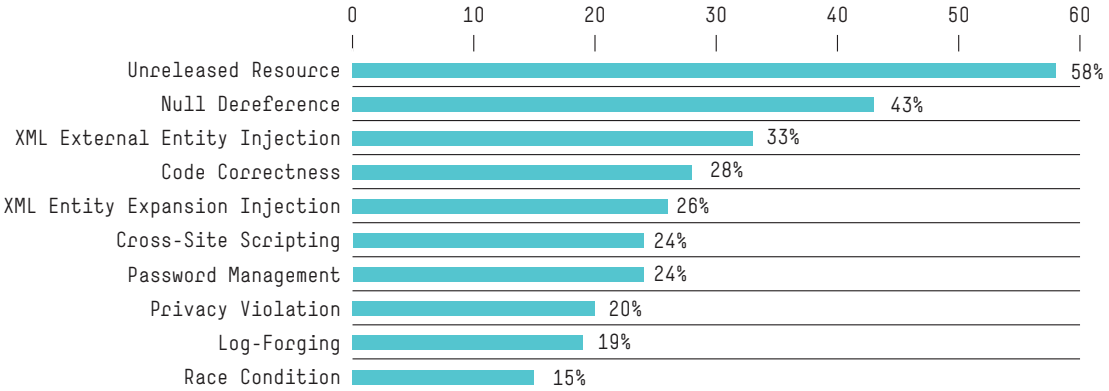


Figure 52. Top 10 vulnerability categories for Java libraries and frameworks in the FOR dataset

The results look very much like those observed in the distribution-by-kingdom data. For example, Code Quality categories Unreleased Resource and Null Dereference are the corresponding number one and two for both Java applications (Figure 50) and frameworks (Figure 52), which meshes with the Code Quality kingdom being the top kingdom across both Java applications (Figure 46) and libraries (Figure 49). Furthermore, as is the case with distribution by kingdom, the percentage of Java libraries that contain Unreleased Resource and Null Dereference issues is slightly higher than those of Java applications, because libraries could provide an API for opening and closing resources, leaving it up to the application to securely use these APIs. Similarly, the leading category for both PHP applications (Figure 51) and libraries (Figure 53) is Cross-Site Scripting, which is one of the most widespread vulnerability categories of the Input Validation and Representation kingdom—the number-one kingdom across both PHP applications (Figure 47) and libraries (Figure 49).

It is interesting to note that the XML External Entity Injection vulnerability category appears in third place for Java libraries (Figure 52) and seventh place for Java applications (Figure 50). We observed similar trends in our analysis of open source Java software dependencies. According to that analysis, XML External Entity Injection tops the list of vulnerability categories for open source Java software dependencies. In general, XML External Entity Injection and XML Entity Expansion Injection vulnerabilities both made the top 10 list for Java applications and libraries. This shows how much Java applications and libraries rely on XML and how much they don't handle it securely. PHP applications and libraries, on the other hand, still struggle with more traditional vulnerability types, such as Path Manipulation, Header Manipulation, Open Redirect, and Cookie Security.

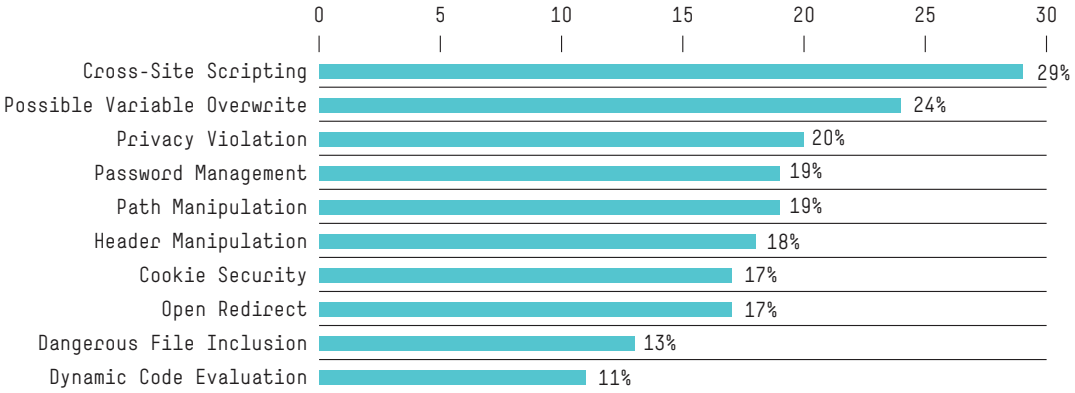


Figure 53. Top 10 vulnerability categories for PHP libraries and frameworks in the FOR dataset

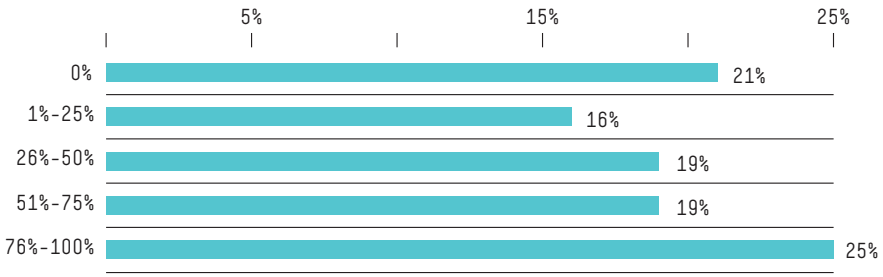


Figure 54. The percentage of open source components in all scanned applications

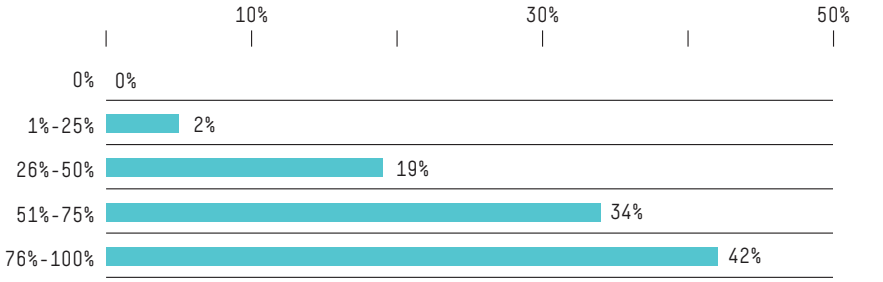


Figure 55. The percentage of open source components in applications new to the dataset in 2015

Open source

Risk analysis of external components

In an assembled-app culture, organizations must keep track not only of vulnerabilities in code developed organically, but also of ones that are consumed as part of referenced third-party libraries. After all, an attacker looks for a hole to allow entry in the organization, and doesn't care how it got there (unless it can lead to other useful points of entry to the targeted system). Last year's Risk Report presented analysis of third-party Java libraries and the known vulnerabilities that those libraries introduced in the applications scanned. This section updates that research, in addition to a few fresh views and insights.

The sample used for this analysis consists of 212 CVEs that were reported across 129 different libraries, if all versions of the same library are counted as a single library. (If each version was to be counted as a separate library, the total is 330.) The usage data was collected from 232 enterprise applications.

Reliance on open source components

Last year, 65% of the applications scanned used at least one open source component. This year that has risen to 79%, substantially due to the apps newly added this year.

Furthermore, 44% of the applications are more than 50% composed of open source components, down from 55% of applications scanned last year.

To further understand the factors contributing to the significant increase in open source component adoption, we looked at open source component usage in the 103 applications that are new in this year's dataset. The distribution of usage is shown in Figure 55.

The Y axis in Figure 55 indicates one probable reason for the increase in open source presence in our dataset—all 103 new applications have at least one open source component. This is another indication that open source components are becoming an integral part of software development. Their weaknesses must thus be taken into account in overall risk analysis.

Input Validation and Representation issues represent the most reported kingdom in Figure 56. The Code Quality kingdom is not heavily represented here, likely because issues in that kingdom tend not to garner CVEs.

Consider a different view of the data, one in which we look at the ranking of the kingdoms when their occurrences across all applications are tallied. This would provide the likelihood of an issue in a kingdom occurring in an application. While Input Validation and Representation and Security Features remain the top kingdoms affecting the applications, in Figure 57 Errors rises up to third place from last in number of CVEs. This shows that just a few issues in the Errors kingdom issues seem to affect 17% of all applications in the sample.

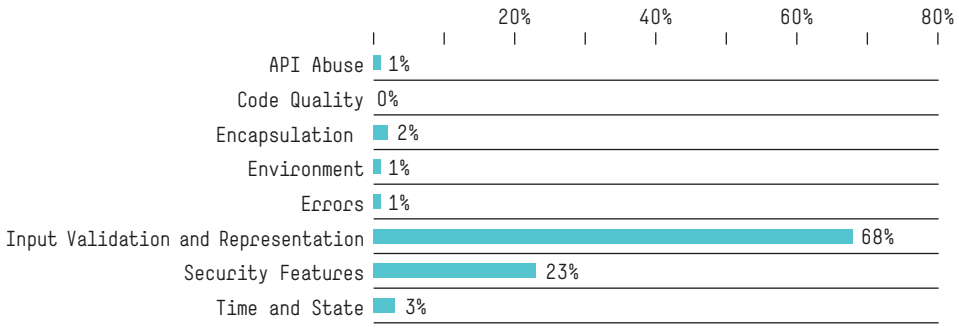


Figure 56. CVEs concerning Input Validation and Representation are the most common type of problems noted in our scans.

Comparing this chart with Figure 48, it's interesting to note that while 77% of applications organically introduced Environment issues, less than 2% of applications inherited these issues from external libraries. This is probably because most third-party references aren't standalone applications, but rather libraries configured from the app itself. Almost 72% of proprietary applications had boundary issues (Kingdom: Encapsulation); meanwhile, 7% of applications have Encapsulation issues because of third-party libraries. For all applications with at least one Security Feature flaw, it is twice as likely that a flaw was introduced organically (that is, by the developers themselves) than that it got into the code by way of a third-party library. Considering that this is the top kingdom of vulnerabilities introduced in proprietary and mobile (Figure 37) as well as open source applications (Figure 48), vulnerabilities introduced due to improper usage (or non-usage) of Security Features seem to plague all types of applications.

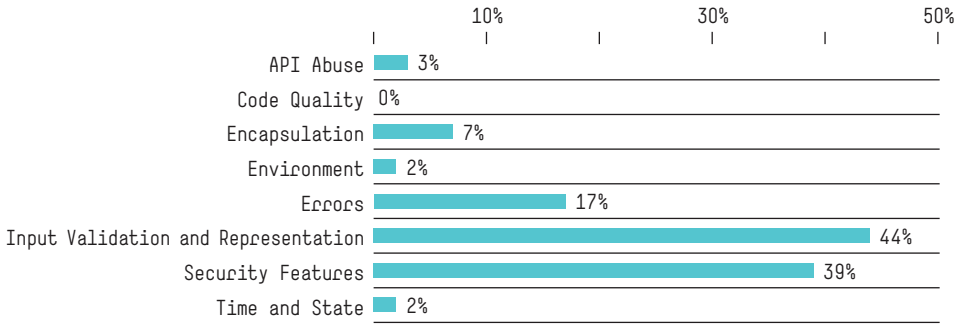


Figure 57. When the count is not restricted to issues with CVEs, issues in the Errors kingdom move up the charts.

XML External Entity Injection has become the top concern this year, switching places with Denial of Service. The vulnerability is pervasive, affecting over 20% of all referenced libraries, which in turn were referenced by 33% of the applications in our dataset. When compared with Figure 59, the top issue not related to Code Quality is in fact XML External Entity Injection (XXE). As mentioned before, code quality issues don't generally get CVEs. That leaves XXE as the most prevalent and disclosed vulnerability in these dependencies during the survey period. This year, fewer encryption-related issues were observed compared to last year, when Insecure SSL was accompanied with weak cryptographic signatures.

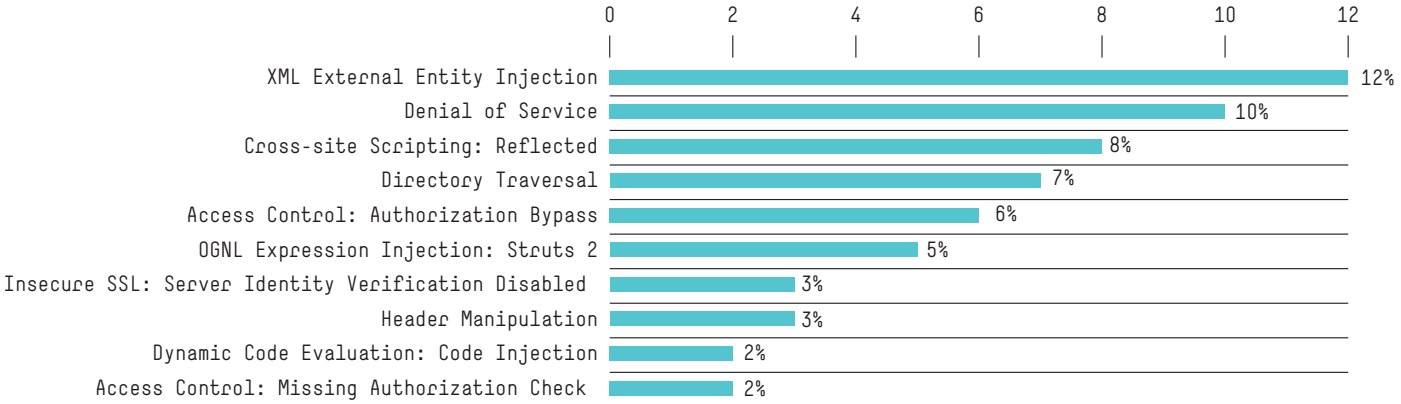


Figure 58. The 10 flaws most commonly seen in our scans, by CVE

Directory Traversal and Header Manipulation made debuts in the top 10 this year. This is an excellent reminder that all data coming into an application must be assumed to be tainted, and should be consumed only after proper input sanitization. As before, one must differentiate between prevalence and severity. Sifting our findings to show only the most severe flaws changes the picture.

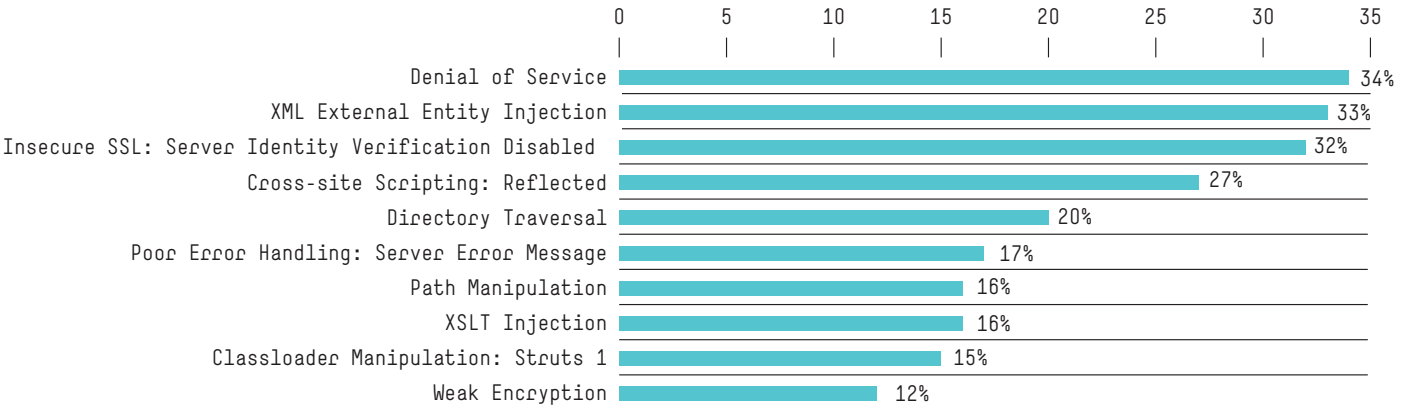


Figure 59. The 10 flaws most commonly seen in our scans, across applications

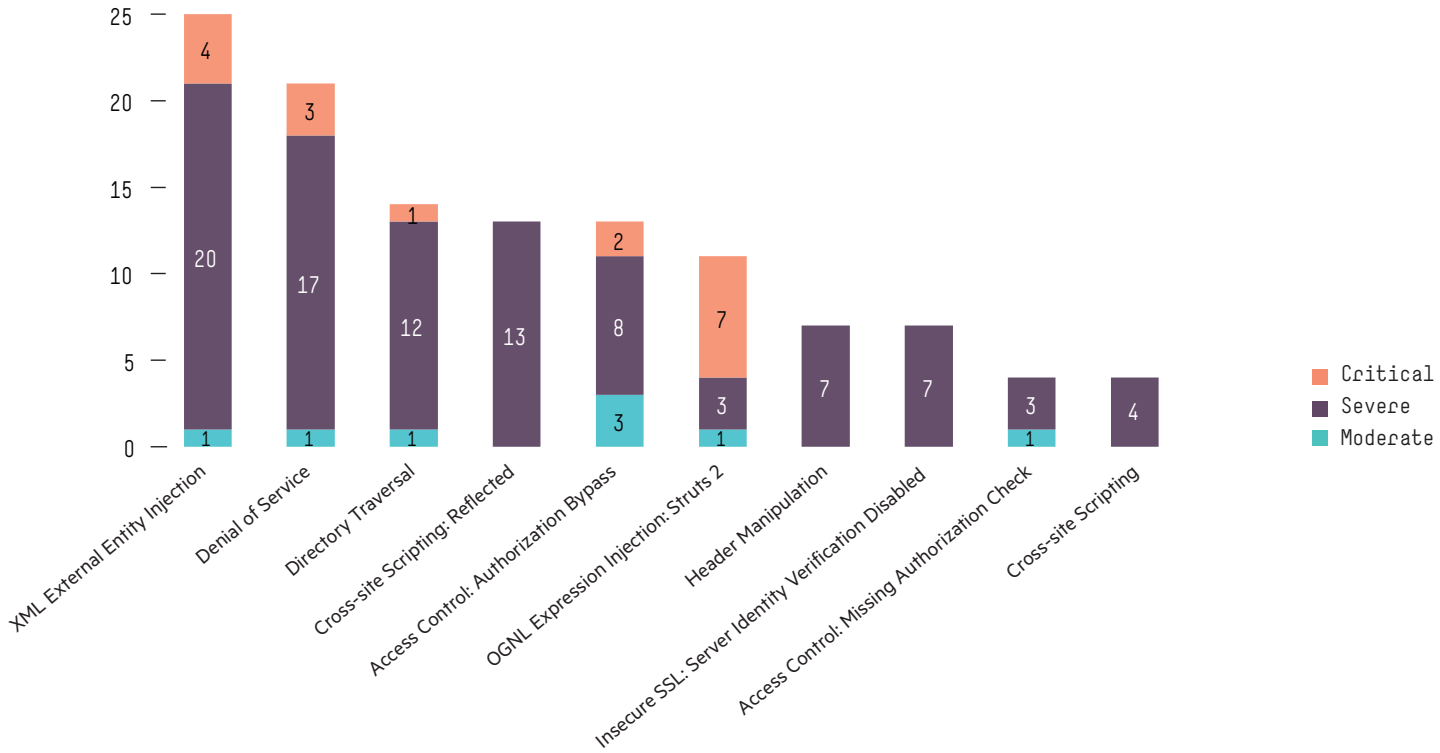


Figure 60. The severity of instances in the top 10 CVEs

It is important to note that although the XXE issue is an Input Validation weakness and normally considered a critical bug, not all instances of reported issues are critical. For the purposes of this analysis, the severity scale is based on CVSS base score and is normalized into three levels (Critical, Severe, and Moderate). Most XXE issues are severe, but only four are critical-class. In contrast, seven out of 11 OGNL Expression Injection instances are considered critical, making it one of the most concerning issues this year. Figure 61 shows the 10 libraries most frequently seen in our sample dataset.

For a glimpse of the complex and fluid relationship between the number of versions released of any particular library and the number of CVEs known for it, take a look at adoption and result patterns for the library most commonly encountered in our scans—commons-fileupload–as shown in Figure 62.

There are seven versions of this library in the dataset, with 1.3 being the latest version. Thirty percent of all applications that referenced open source components in the sample referenced some version of the library. However, only four referenced the latest 1.3 version; of those four, only one had upgraded to the latest from an older version.

Generally speaking, about 9% of all scanned applications that use at least one open source component in the sample upgraded at least one of their libraries. Within this data subset, only around 5 percent of applications upgraded to the latest version of the library.

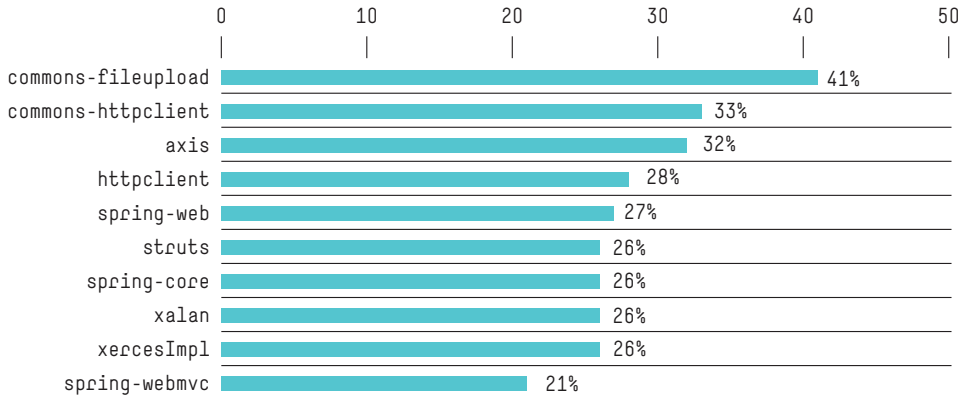


Figure 61. The most popular libraries seen in the dataset

Overall, 49% of applications referencing open source components used the latest version of some library.

Lastly, we looked again at the five most vulnerable applications scanned in 2015 in order to compare the vulnerabilities found in each application's native code to the vulnerabilities each inherited from external dependencies.

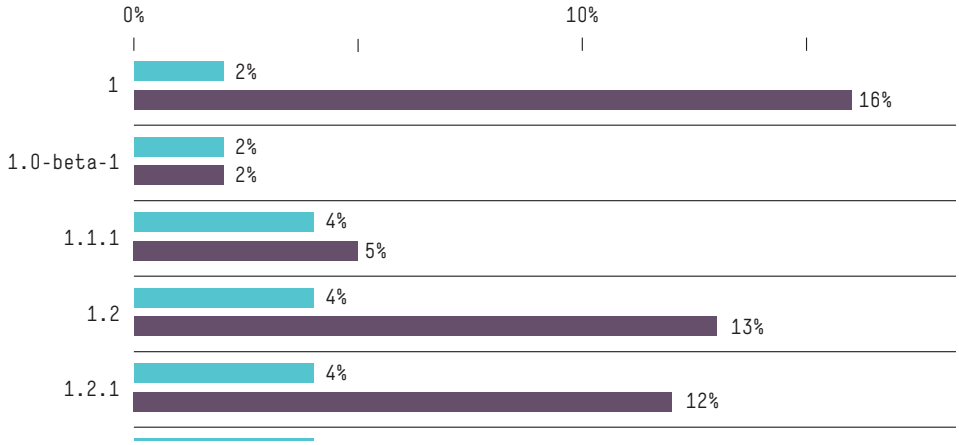


Figure 62. Commons-fileupload: portrait of a popular open source library

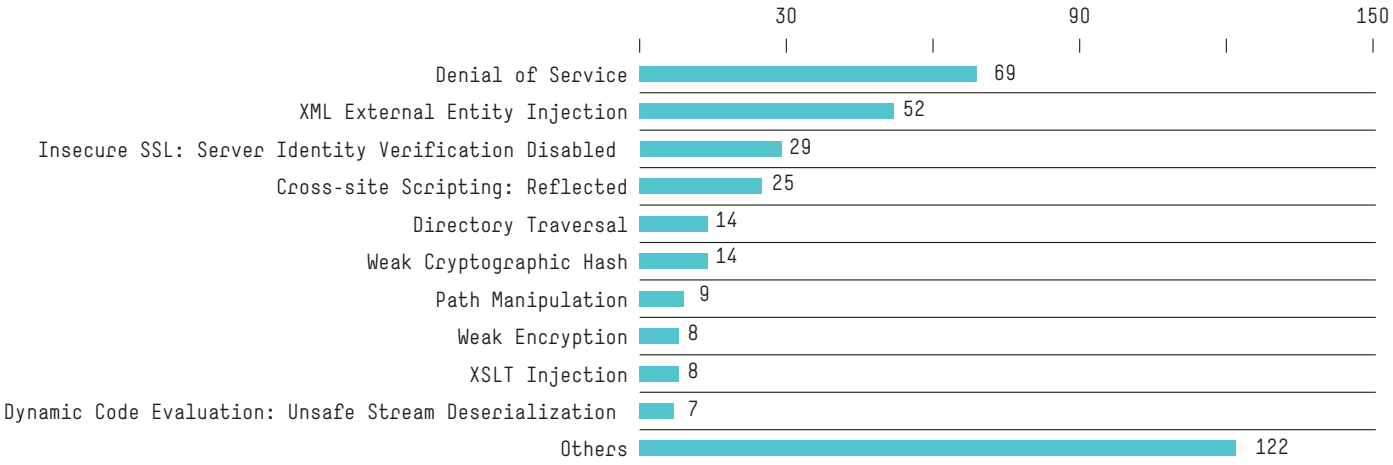


Figure 63. CVE-level issues inherited from external dependencies by the five most vulnerable applications of 2015

Compare Figure 63, which shows vulnerabilities inherited from external dependencies, to Figure 64, which shows vulnerabilities occurring in each application's proprietary code.

As the charts show, there's not a great deal of issue overlap—only two of the top 10 issues (Cross-Site Scripting: Reflected and Path Manipulation) appear in both charts. It's also notable, and concerning, that Null Dereference is one of the top issues for proprietary code. Null dereferencing can lead to severe security consequences including, but not limited to, root exploits. However, such issues often fail to garner enough attention in-house to be averted.

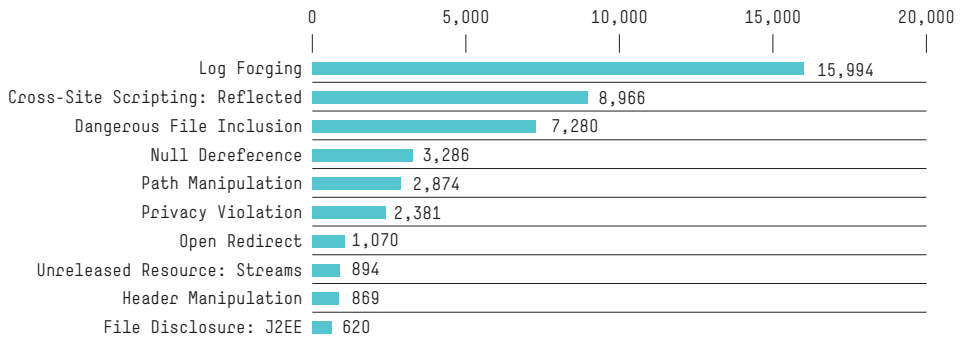


Figure 64. Issues found in the native code of the five most vulnerable applications of 2015

Remediation

We next look at this year's data concerning remediation rates. For this analysis, we looked at vulnerabilities that were both found for the first time and fixed within the same yearlong period (October 30, 2014, to October 30, 2015). All vulnerabilities represented in the data were triaged and closed. The closed issues may or may not have been remediated. This question is considered in the analysis below.

Number of vulnerabilities fixed

We begin with the applications dataset. Figure 66 shows the percentage of vulnerabilities fixed across all kingdoms, color-coded by severity.

Overall, more than 92% of all issues closed were remediated. This is good; most of the issues that are triaged are seen through to remediation. The remediation rate in Errors is particularly fine, perhaps because problems in that kingdom are generally easier to fix. In contrast, the API Abuse kingdom appears to be lagging. Further data analysis indicates that the anomaly can be ascribed to a very few applications—less than 2% of those in the dataset—that did not successfully remediate three specific types of weakness: Mass Assignment, ASP .NET MVC Bad Practices, and File Disclosure. Shifting to mobile applications in Figure 66, it's important to note our mobile remediation sample size was extremely limited, especially compared to that available in the applications sphere. Though the sample is more or less evenly balanced between Android and iOS applications, the total number of samples in mobile (45) is such that our researchers regard the data here as interesting, but possibly non-representative of the wider world.

Only 48% of the mobile issues in our sample seem to have been remediated—a stark difference from the 92% we saw on the applications side. There are a couple of anomalous moments in Figure 66, as one might expect from a very small dataset. For instance, the poor showing in Environment is actually down to one specific Android app with multiple issues in that kingdom. The developers of that app did not successfully remediate, and the bar chart pays the price (as do, presumably, the users). In this dataset, 86% of all vulnerabilities fall into the Security Features, Code Quality, or Environment kingdoms. In the Security Features kingdom, 57% of all issues are in the Privacy Violation category, out of which around 53% were remediated. In turn, within Privacy Violation, more than 65% of the findings were Screen Caching issues, and 54% of those were remediated. Alas, almost all of the issues not remediated were critical- or high-severity issues.

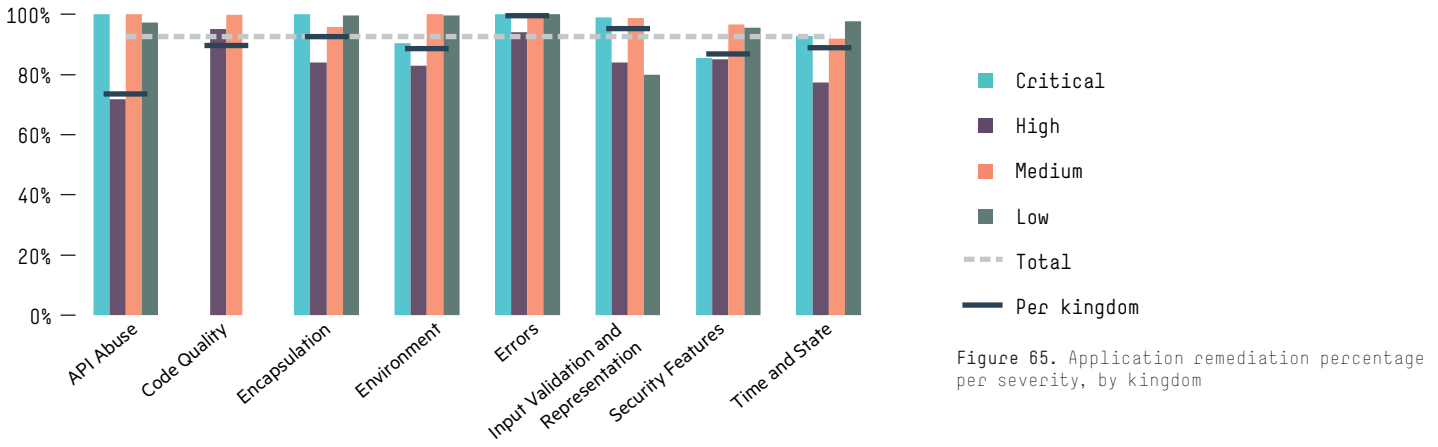


Figure 65. Application remediation percentage per severity, by kingdom

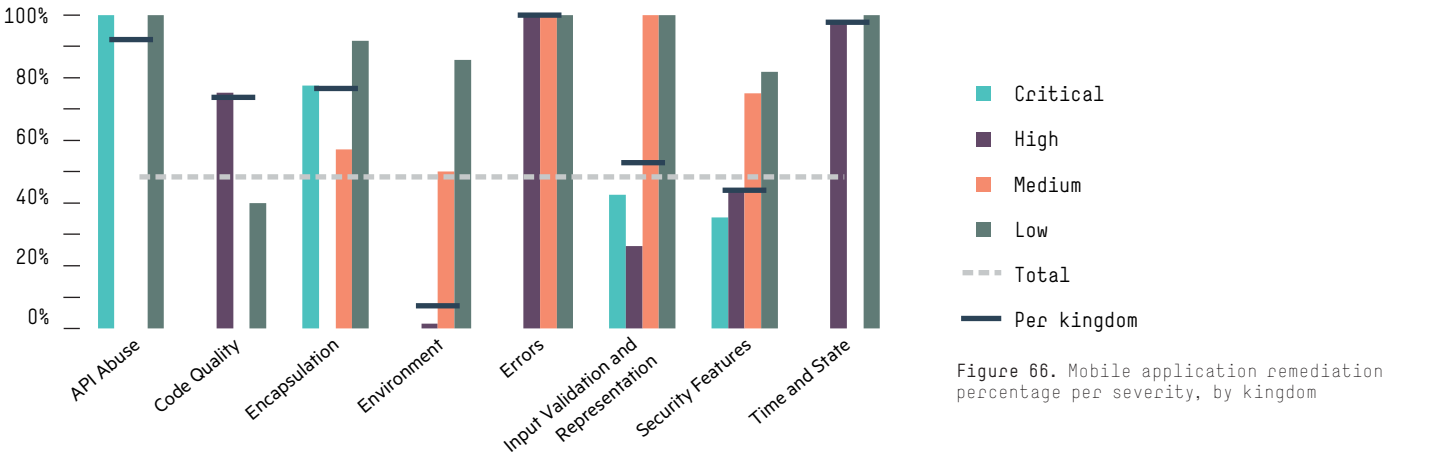


Figure 66. Mobile application remediation percentage per severity, by kingdom

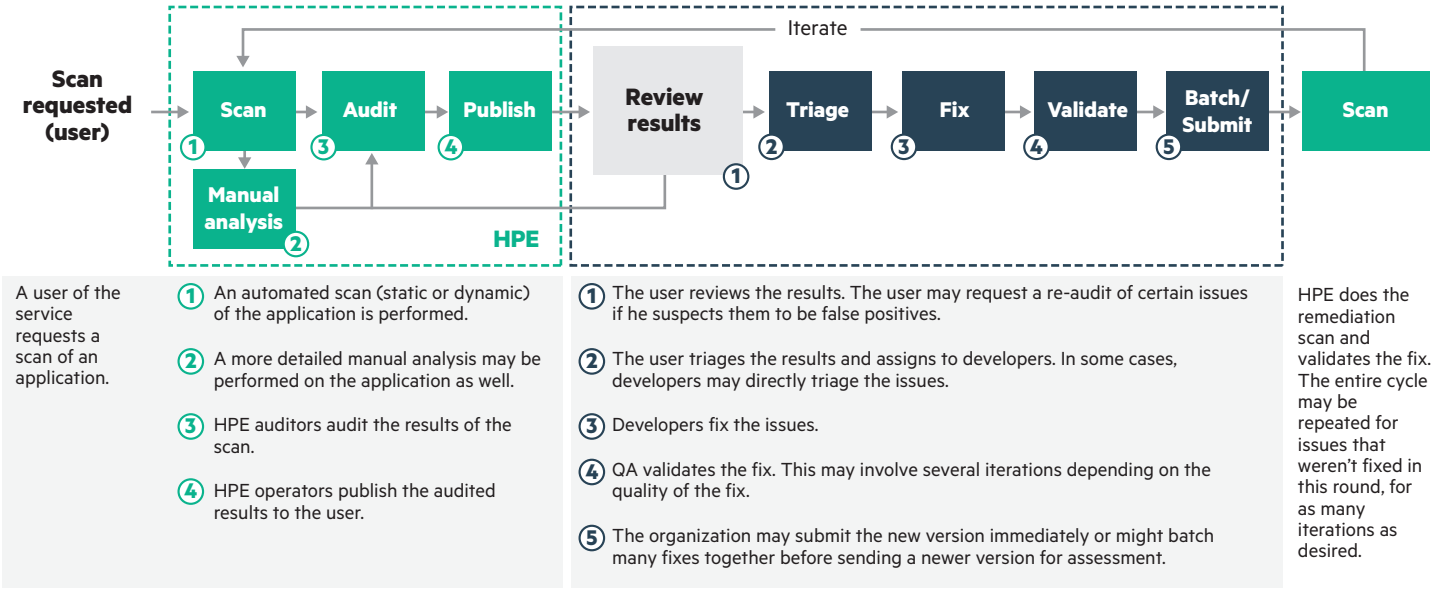


Figure 67. Remediation process

Remediation: How the process works

What happens between the moment an application first reveals its vulnerability to a Fortify scan and the next time it meets the scanner? Typically, the process unfolds like so:

The scanning patterns of users for static and dynamic scans vary. Static scans are usually more frequent, with a median of six days between scans. There tends to be a longer interval—27 days—between dynamic scans. Due to this variation, the following analysis is performed separately for static and dynamic scans within our sample set.

Scan results

In order to analyze remediation patterns among static scans, we compiled a sample dataset of 327 applications.

Most low-severity vulnerabilities found in static scans seem to be fixed very early on. In this dataset, these numbers reflect System Information Leak issues, most of which were relatively straightforward and fixed very quickly. Most critical-severity issues are addressed in the second range of scans (6 to 11 scans, or 31 to 60 days). This may be because critical vulnerabilities tend to require longer investigations, and the fixes take longer to engineer as well. About 77% of the cross-site scripting issues we saw were fixed in this range. Of those, the vast majority were in the Cross-Site Scripting: Reflected category, though a few Persistent and DOM issues turned up as well. This pattern makes sense due to the general pervasiveness of Reflected XSS compared to other examples of the problem.

Further down the chart, the random spikes after the fourth range of scans are caused by unusual activities in a very few applications. These are anomalous artifacts of the small dataset and do not represent the pattern exhibited by a majority of the applications.

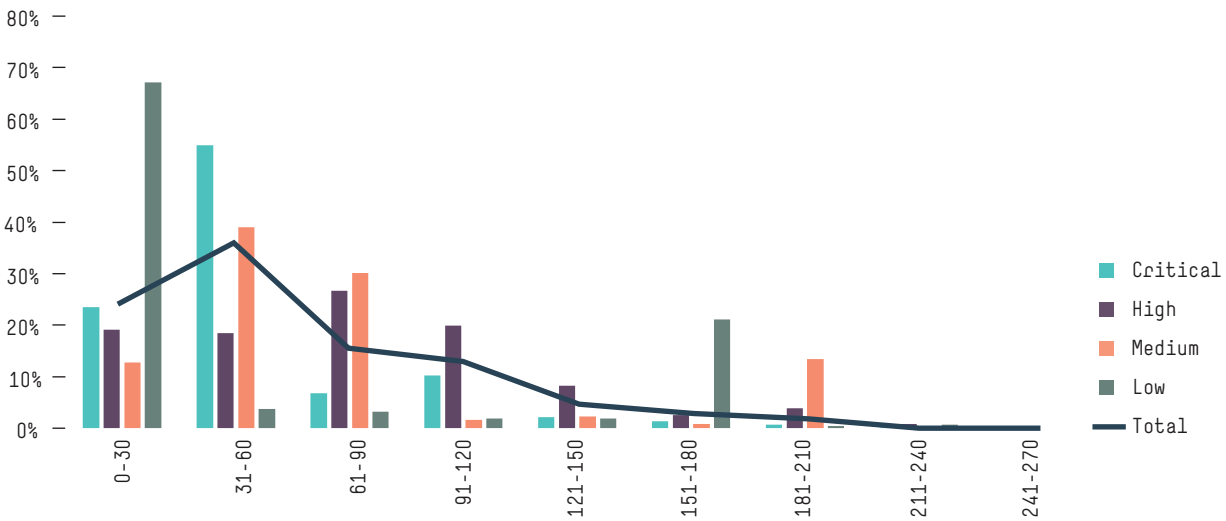


Figure 68. Remediation patterns in static scans of applications

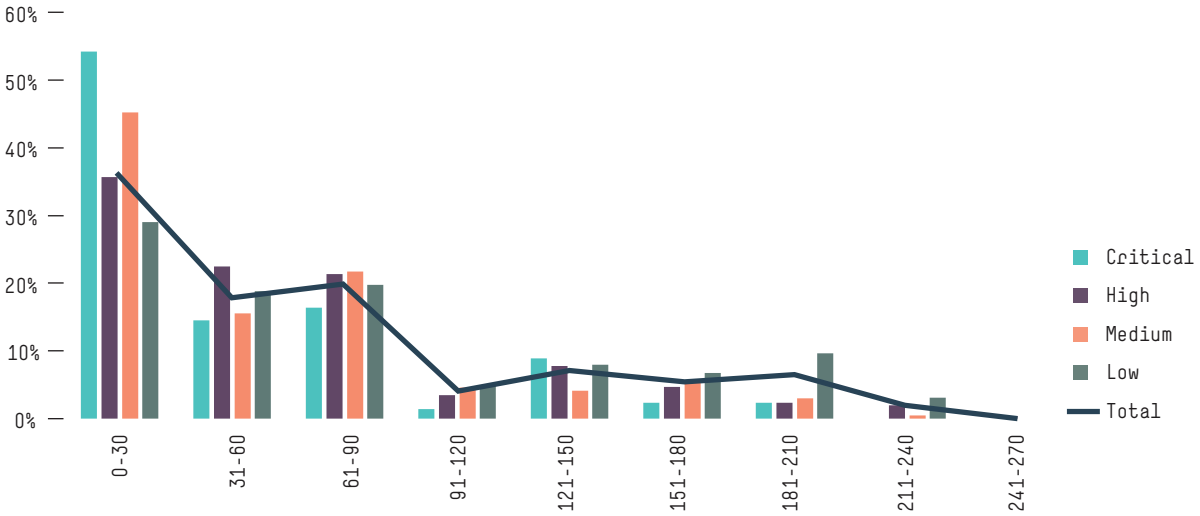


Figure 69. Remediation patterns in dynamic scans of applications

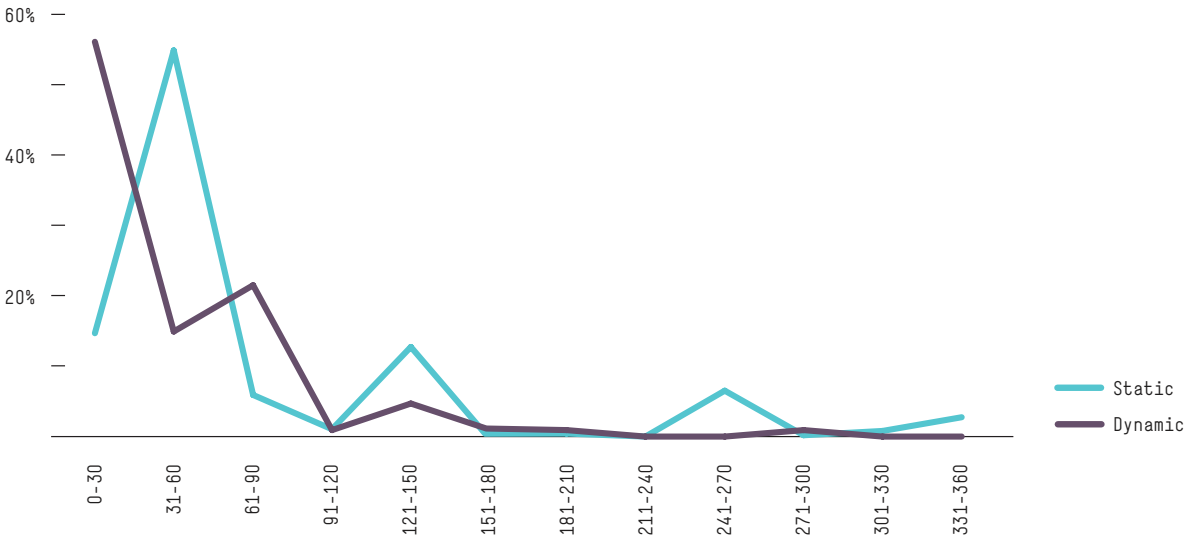


Figure 70. Remediation patterns in static and dynamic scans for mobile applications

We used scan data on 301 applications to analyze remediation patterns among dynamic scans. In Figure 69, each range roughly corresponds to the time between two dynamic scans. A lot of vulnerabilities of every severity are addressed early on. Of particular note, most critical-class Cross-Site Scripting findings were remediated within the first range of dynamic scans. In contrast, the critical XSS findings caught by static scans weren't remediated until the second scan range (Figure 68). The difference may lie in the kinds of information presented to developers by the two scan technologies.

Turning our attention to scanning patterns for mobile, we noted that the median time between scans was much closer—17 days between static scans, 21 days between dynamic scans. Once more, this may be an artifact of the small dataset, or something else may be a factor here. It makes sense to present these numbers in a single chart.

Although the sample dataset is much smaller than that available to examine remediation of application vulnerabilities, it is still interesting to note that overall trends of dynamic and static scans over 30-day ranges are very similar to those observed with the application scans.

With all this analysis, we're now in a position to make some cumulative observations about our remediation data.

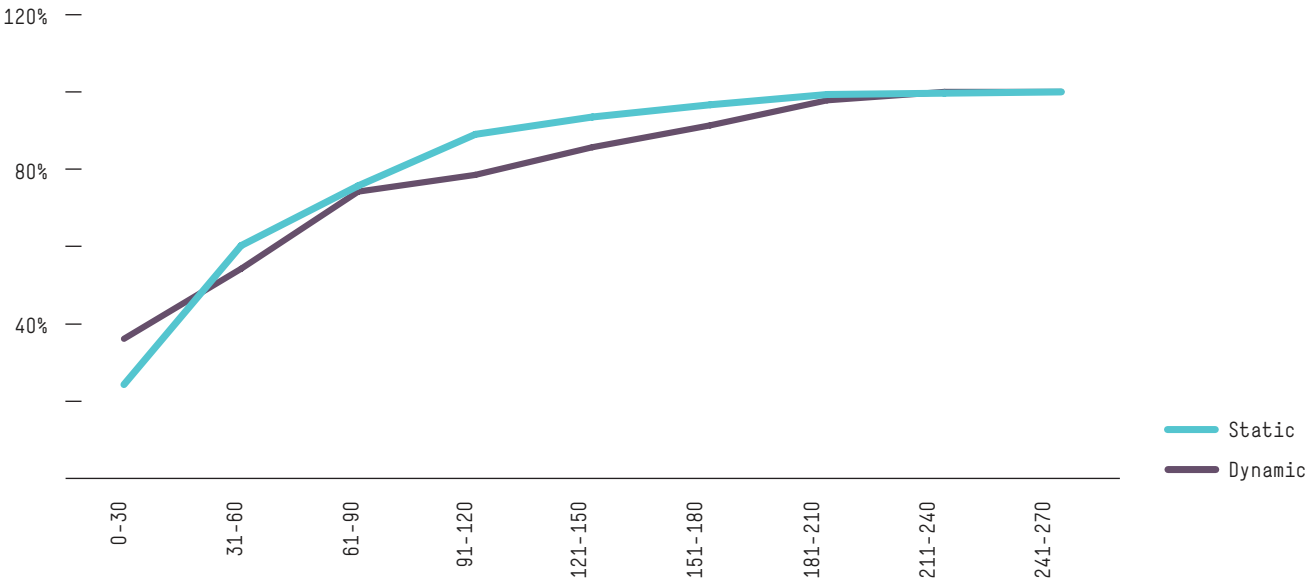


Figure 71. Cumulative remediation of issues over time for applications

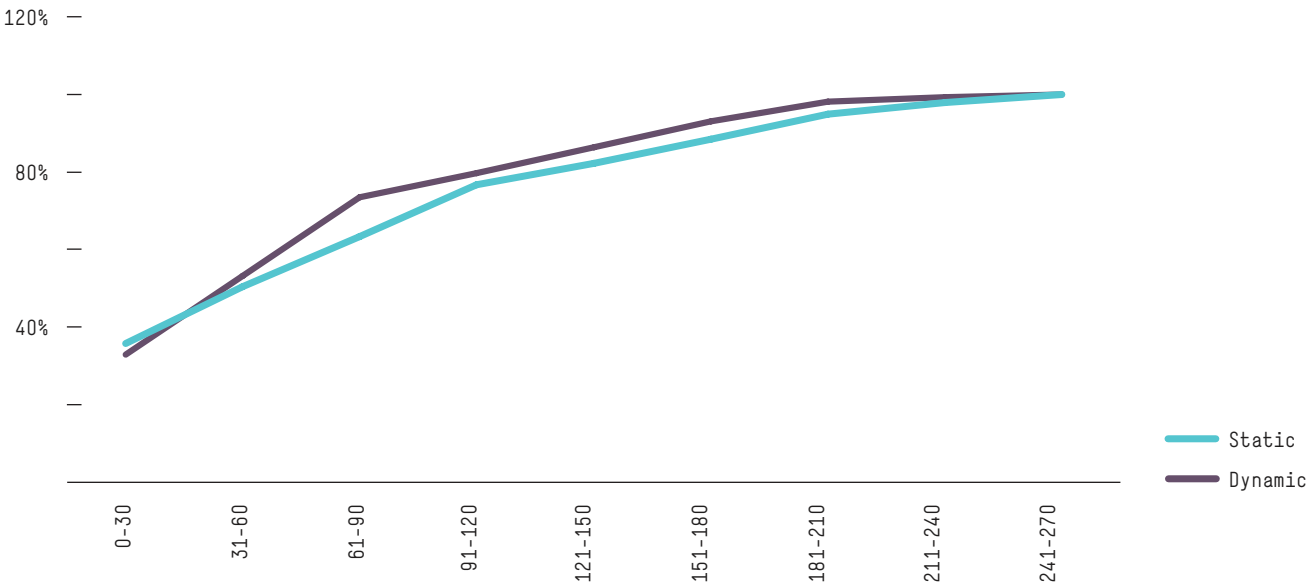


Figure 72. Cumulative remediation of remediated applications

Overall, about 90% of all issues discovered in static scans seem to be resolved within the first four ranges. Within those first four ranges, we saw spikes for each severity of vulnerability. This shows that vulnerabilities of all severities are addressed within this time, and that outliers to those ranges are not determined by severity class.

Looking at the speed with which application developers remediated their triaged issues, there is notable parity between scan technologies. For static scans, 35% of all applications remediated issues that were spotted within the first scan range (that is, within six static scans; the day of the first scan in which the vulnerability appears is numbered as day zero). By the end of the fourth range (21 scans, or 90-120 days), that percentage was up to 76%. For dynamic scans, 32% of issues were remediated within the first range. By the end of the fourth range, the percentage was up to 79%. Though there's certainly a functional difference between the static and dynamic scan technologies, both are clearly being used for their intended purpose of making apps better.

Finally, let's see how mobile time to fix shapes up, keeping in mind once again the oddities a small dataset brings.

In this sphere, vulnerabilities reported from dynamic scans seem to be remediated a bit faster than those from static scans, though again the difference in static and dynamic scan intervals on the mobile side is not as great as the gap on the applications side.

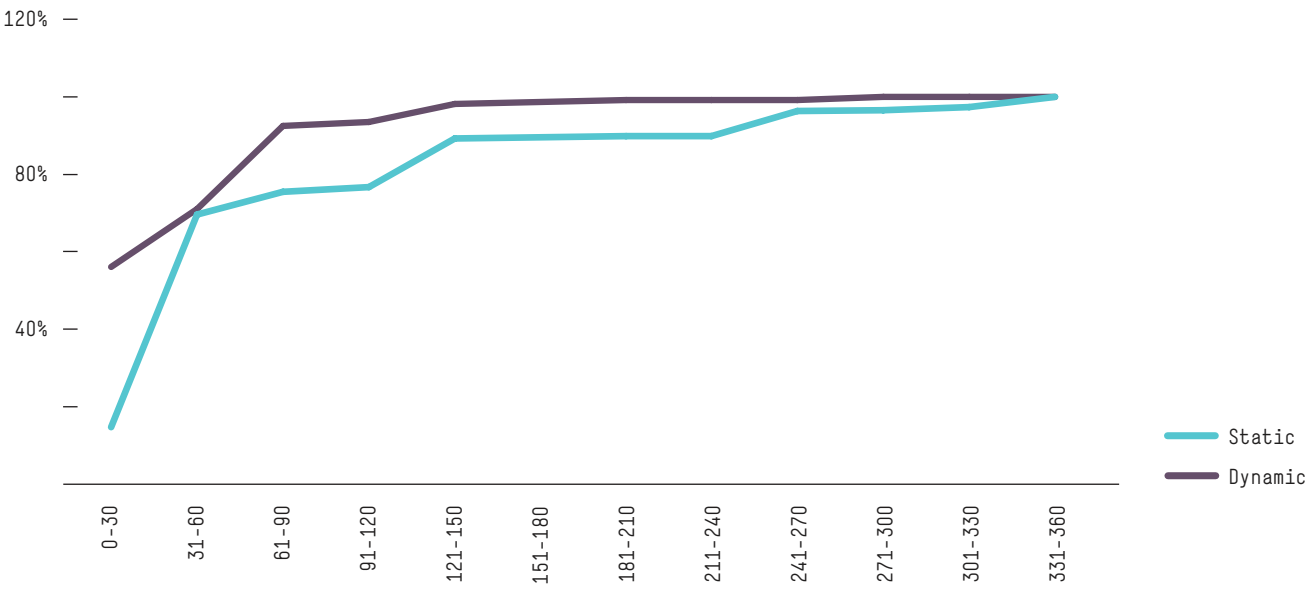


Figure 73. Cumulative remediation of remediated mobile applications

Conclusion

Overall, it has been an interesting year for software security research. Both applications and mobile software pose unique challenges to developers, and various vulnerabilities detected in these platforms support that impression. It was also interesting to note that applications and mobile shared certain trends in vulnerabilities when analyzed by kingdom, thus pointing to common fundamental failures in the software. The rate of vulnerability remediation seems to be increasing, which suggests that technologies are becoming better understood as they mature. Nevertheless, there is room for improvement as shown by the prevalent issues detected.

On the mobile front, the race to compete in a new, powerful market has forced vulnerable deployments with known issues. Moreover, all types of applications tend to use third-party libraries to ease and speed up the development process. But such actions can lead to inheritance of additional vulnerabilities from yet another source. Here, two main items need to be noted. While most high-impact issues in third-party libraries are disclosed as CVEs, it is disturbing to note that the applications that use them are not updated soon enough. Also, CVEs do not represent all the issues found in third-party software and, as shown by data from the FOR project, other undisclosed issues may still exist. Based on these discoveries, more awareness and training could be offered to improve the quality of applications being developed. The goal of secure software development is not only to remediate all vulnerabilities, but to develop applications that don't have vulnerabilities in the first place. While we move toward this ideal world, the right kinds of investment could take us closer to the goal.

Defense and defenders

The security state of defenders

As organizations struggle to address security gaps and to operate in an assume-the-breach world, defenders grapple with the need for improved detections. Currently, the most common method of event detection involves monitoring correlated log data, but there's a whole world of options for defenders to navigate. For this year's Risk Report, we polled defenders and derived a clearer picture of the defender landscape.

All organizations need the ability to respond to threats. In research we conducted[299] among a self-selecting group of incident responders and enterprises, 80% of respondents report having security operations functions within their organization as seen in Figure 74.

[299] www.surveymonkey.com/r/ProtectSOC.

Fifty-one percent report the presence of a formal security operations center (SOC), and almost 11% reported themselves to be in the process of building one, as seen in Figure 75. These numbers do not take into account small or midsized businesses[300] in which the expectation would be a much smaller percentage of formal SOCs.

The fundamental purpose of a security operations center is monitoring. Additionally, according to Ponemon's 2015 Cost of Cyber Crime Report, detection and recovery make up 53% of internal activity cost (incident/non-budget costs), followed closely by containment and investigation—all processes that are often managed by security operations.[301]

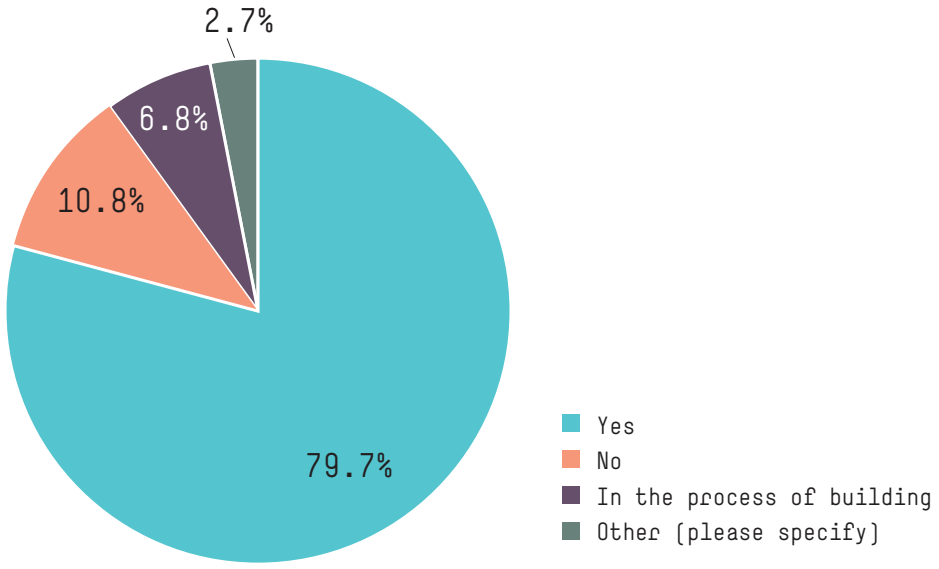


Figure 74. Responses to the question, "Does your organization have a security operations function?"

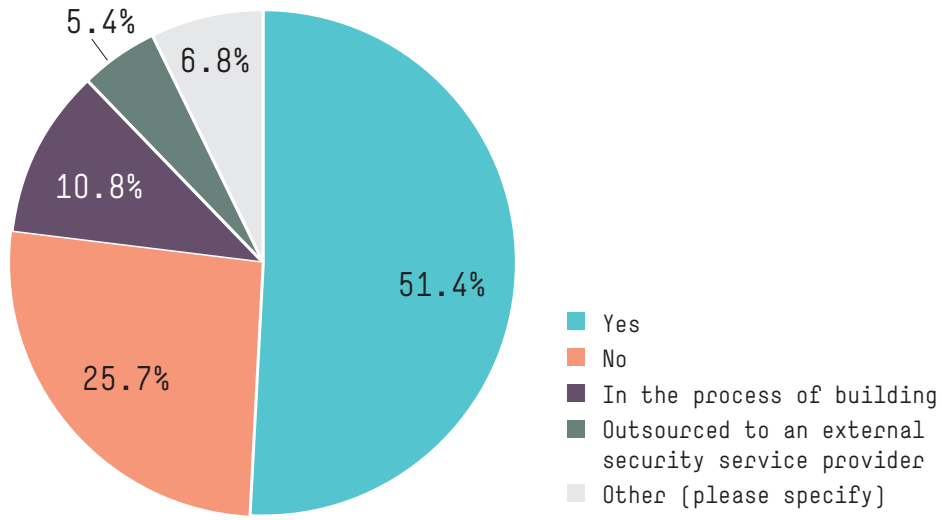


Figure 75. Responses to the question, "Does your organization have a security operations center (SOC)?"

[300] http://www.gartner.com/it-glossary/smbs-small-and-midsize-businesses.

[301] http://www.hp.com/go/ponemon.

Answer options	Response percentage
IPS/IDS	85.5
Proxy	72.7
Netflow	47.3
Firewall	74.5
WIDS/WIPS	23.6
AV/HIPS	67.3
Server (RHEL, Windows Server®, etc.)	60.0
Internal applications (SAP, CRM, HR, etc.)	27.3
Authentication (Active Directory, LDAP, etc.)	74.5
VPN	67.3
DLP	27.3
PKI	25.5
DNS	65.5
Database (Oracle, SQL Server, MySQL, etc.)	45.5
Web applications	41.8
Cloud services (Azure, Google, AWS, Adallom, Office 365, etc.)	16.4
Other	5.5

Figure 76. Responses to the question, "Please select all data sources regularly monitored by security operations (SIEM/SOC)."

Four blocks to implementation

In security operations, the reactive nature of security monitoring is commonly the subject of complaints. Events must occur before they can be detected, as opposed to the more proactive prevention approach. Of course, the reactive-proactive conversation must occur within the context of the technology available, the most common of which is a security information event management system (SIEM). Interestingly, in our research we found the frequency of SIEM implementation fell between those entities with a SOC and those with an operational function. As identified in the Ponemon Report, "the use of security intelligence systems (including SIEM)... translates to an average cost savings of $1.9 million."[302]

Operations analysts' ability to detect an event is predicated on their ability to see relevant event data. While SOC nirvana would mean real-time detection and analysis, there are several concerns that may affect implementation of real-time monitoring.

Types/Lack of events. As identified in the summary of the Ponemon report, most organizations still spend about 30% of their security budget on the network layer. The implications of this become clear when reviewing the types of logs organizations actively monitor, as we see in Figure 76.

Staleness of data. While 36% of respondents claimed real-time ingestion of event log data, almost 50% admitted to a mix of real-time and batch data, as shown in Figure 77. As an example, one accounting firm that responded to the survey chose to use the batch setting, instead of a real-time stream, in its high-volume proxy devices. This decision reduced the load on the proxies, but created a window of up to six hours on proxy log events. While the time difference between event time and SIEM receipt time is displayed, if analysts aren't careful in that situation they could inadvertently find themselves investigating activity from several hours back. To avoid this, content must be written to correlate with events received asynchronously.

[302] Op. cit.

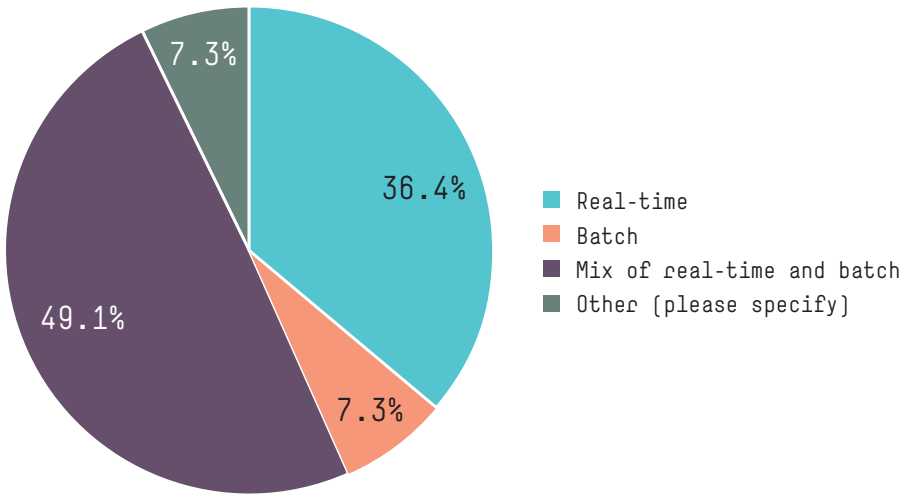


Figure 77 Responses to the question, "What best describes the timing of logs and security events into your SIEM for SOC monitoring?

Environment event coverage. For decades, operational logs have been collected to facilitate troubleshooting. However, in an effort to maintain availability, the collection process has relied on the centralization of IT. The largest gap in security log collection occurs in areas where operational log collection has not been a priority. In many organizations, server log collection is limited to events dictated by various compliance requirements, or is stymied by a lack of centralized management and concurrence in event collection. Likewise, client host event collection is virtually non-existent, with ROI decisions focused on the cost to re-image machines versus event collection infrastructure, storage, and even troubleshooting investigation costs. Antivirus data is one of the primary client and server host findings most organizations collect and feed into their SIEM, although log collection and monitoring infrastructure scale concerns grow as the company does.

Security analysis content. For security operations to have a chance of analysis and detection of intrusion, SIEM content must exist that allows reasonable notification of security relevant concerns. The creation and honing of content is more than a data problem; it's simple math. Let's assume an organization has 20 different device types to monitor (firewall, server, proxy, antivirus, applications, and so on), and each of the device types has the potential to generate 200 different events. This means we have a data field of 4000 potential event IDs to review. Some events will be strictly operational, some will be security, and some could apply to both, so conservatively we reduce the interesting event types to 75 per device, yielding an analysis field of 1500 interesting items. Now take this base of 1500 interesting events and multiply it by 500 individual devices—and don't forget that this data expands exponentially when factors such as device OS versions and time are included in calculations.

These equations become important when considering security information detection and correlation. Some may believe reactive security operations to be at a disadvantage, as they must spot the initial intrusion to effectively protect an organization. In reality, however, this is inaccurate; there are multiple points at which defenders can detect, and contain, an issue. Monitoring may catch an event at any point in the attack chain, at any device event, at any time there is malicious activity. The data to be considered expands considerably, but so does the ability to detect. There is not one single event in an active intrusion, but the potential to catch any activity that occurs.

Analysis of breaches over the last couple of years shows a major trend of attacks that focus on applications and their assets.

OpSec: detection in the real world

The 2014 Ponemon report noted that "certain organizational factors reduced the overall cost [of a breach]. If the organization has a strong security posture or a formal incident response plan in place before the incident, the average cost of a data breach was reduced by as much as (respectively) $21 and $17 per record. Finally, appointing a CISO to lead the data breach incident response team reduced per capita cost by $10."[303] While there's not a direct correlation between employing a CISO and reducing breach costs, one need look no further than the 2014 quarterly profit expectations at Target, an organization without a CISO at the time of its breach.[304]

Let's look at how detection played a role in the January 2015 Anthem data breach, which we mentioned in the privacy section. When Anthem reported the incident,[305] it had been underway for seven weeks—well below the average breach duration of 200+ days for the dataset cited in Ponemon.[306] (Other studies suggest other durations.[307]) The company figured out it had been breached when a sharp-eyed database administrator noticed unusual activity—specifically, a database query made with his ID code.[308] This all sounds very positive, but Anthem probably didn't feel that way, as it was one of the largest known corporate breaches of personal information to date (at that time).[309] It was also Anthem's second time at the breach rodeo. It was compromised as well in 2010 under a previous name, WellPoint.[310]

The Anthem breach is just one of many indications that attackers have shifted their efforts to directly attack applications. Another example might be the Patreon breach, in which attackers utilizing a SQL injection

flaw made off with not only user data but the service's source code, presumably to scan for more vulnerabilities at their leisure.[311] Unfortunately, a business' efforts to improve accessibility and ease of use can increase ease of access for attackers too. Those who monitor the attack marketplace have confirmed the increased presence of application exploit toolkits there.[312] And analysis of breaches over the last couple of years shows a major trend of attacks that focus on applications and their assets.[313]

In the process of making it easier for customers to do business, companies have unwittingly given attackers greater access to the center of their business, which is typically a suite of applications wrapping their business data and processes. The success of these attacks in spite of perimeter, network, and traditional application defenses indicates that defenders must adapt their strategies.

[303] http://www.hp.com/go/ponemon.

[304] http://money.cnn.com/2014/05/21/investing/target-earnings/index.html.

[305] https://www.anthemfacts.com/.

[306] http://www.hp.com/go/ponemon.

[307] http://www.scmagazine.com/companies-leaving-known-vulnerabilities-unchecked-for-120-days-kenna/article/441746/.

[308] http://www.wsj.com/articles/health-insurer-anthem-hit-by-hackers-1423103720.

[309] http://www.ibj.com/articles/51789-anthems-it-system-had-cracks-before-hack.

[310] https://www.duosecurity.com/blog/four-years-later-anthem-breached-again-hackers-stole-employee-credentials.

[311] http://arstechnica.com/security/2015/10/gigabytes-of-user-data-from-hack-of-patreon-donations-site-dumped-online/.

[312] https://www4.symantec.com/mktginfo/whitepaper/ISTR/21347931_GA-internet-security-threat-report-volume-20-2015-appendices.pdf.

[313] http://f6ce14d4647f05e937f4-4d6abce208e5e17c2085b466b98c2083.r3.cf1.rackcdn.com/work-smarter-harder-to-secure-your-applications-pdf-9-w-1884.pdf.

Direct defense and automation

Organizations are becoming more aware of the need for direct application monitoring and defense. In our recent survey[314] of organizations, 28% said they monitored their internal applications for security-related events, and 43% reported monitoring their external-facing applications. This is a dramatic increase from just a few years ago, when almost no one was monitoring applications for security-related events. Industry analysts have also recognized[315] new product market categories for direct application monitoring and defense, with most products allowing for centralized monitoring and analysis.

Whether related to a direct application attack or simply a vulnerability, studies have shown a security flaw will most likely be exploited within 60 days of discovery, but companies may take over 100 days to remediate it.[316] This poses the question, "Can humans react fast enough to network security attacks?" Most of information security history points to the answer being "No." This is borne out by the great number of successful attacks that were detected, some even at the early stage of the attack, but that companies nonetheless failed to stop in time.

Many companies are relying on installing more detection systems, many of which are placed in passive mode, meaning they will not interrupt traffic when a threat is detected. This is primarily because one device is not enough to differentiate between an actual attack and a false positive.

One of the manual methods used to mitigate a confirmed attack is to configure a firewall to stop the traffic. The problem is that these configurations take time to implement, and the attack may have already succeeded in removing data or compromising systems. Undetectable malware may have been deposited to wait and send data out to one or more Internet hosts. Knowing your network, addressing, and current real-time network configurations along with point-of-contact admins along each node helps to cut down on response time.

The industry generally agrees that the fastest remediation is automated remediation.[317, 318] For example, defenders might use a security enterprise manager to correlate events from different devices, first confirming an actual attack is occurring, then sending commands to routers or firewalls to block the traffic.[319] This may cause some inconvenience to company users, but the alternative, as Anthem and so many others can attest, is much worse.

Our research found a relatively low degree of comfort with application monitoring. The trend of using applications directly for threat intelligence is new, and many of the organizations surveyed did not know exactly what applications to monitor or how they should monitor their applications, or what security threats could be derived from this data. This low visibility, combined with the fact that application-related breaches are not decreasing despite the use of more traditional security methods, shows that this area of security intelligence is still immature. Most organizations are not yet getting the results they need and are not keeping pace with attacker trends. But defenders must adapt to survive. They must learn to treat applications as security devices. An application should have defensive capabilities, and it should provide security-related events such as authentication, authorization, configuration, and resource access to security analysts.

[314] www.surveymonkey.com/r/ProtectSOC

[315] https://www.gartner.com/doc/3090717/hype-cycle-application-security.

[316] http://www.complysmart.com/component/easyblog/entry/study-claims-enterprise-vulnerability-remediation-can-take-120-days.html?Itemid=52.

[317] https://research.gigaom.com/report/intelligence-aware-threat-detection-and-mitigation/.

[318] https://www.qualys.com/docs/guide_vulnerability_management.pdf.

[319] http://cyberattackdefenders.com/blog/security-operations-center-soc-automation-why-it-matters/.

Conclusion

In research we conducted among a self-selecting group of incident responders and enterprises, four-fifths of respondents report having security operations functions within their organization. This low number gives us pause, because it is clear from our research that many organizations are not keeping pace with attacker trends, including direct attacks of the systems on which enterprises rely. We found evidence that adversaries are taking excellent advantage of technologies enterprises have put in place to serve their customers. Only by learning to treat applications as security entities on the network can defenders hope to adapt to the new adversary landscape.

Trends in security: the conference scene

Despite the wealth of research resources available to us in-house, the HPE Security Research team still remembers there's a world outside its doors. Last year, our researchers looked at security trends in the news. This year, we turned our attention to what drew interest at information security conferences in 2014 and 2015.

We discerned a number of security trends, based on analyzed abstracts for 4239 accepted talks at more than 80 industry-oriented conferences and six top-tier academic security conferences. Because a single talk can touch on more than one theme, Figures 78 and 79 depict the overall distributions for 12 security tracks that cover almost every aspect of information security. (Note that we determined some talks to fit into more than one category.)

In the industry sphere, threat and vulnerability topics were the meat of the conference scene, with talks in this category appearing in one-third of the abstracts analyzed. Privacy continues to excite discussion, as do GRC (governance, risk, and compliance) and mobile topics. Despite high levels of industry chatter about the Internet of Things, IoT-related topics made a weak showing in both 2014 and 2015, though both IoT and mobile saw slight bumps of a percentage point year over year. Privacy, network security, and data security itself declined in interest to conference organizers as did, surprisingly, cloud security.

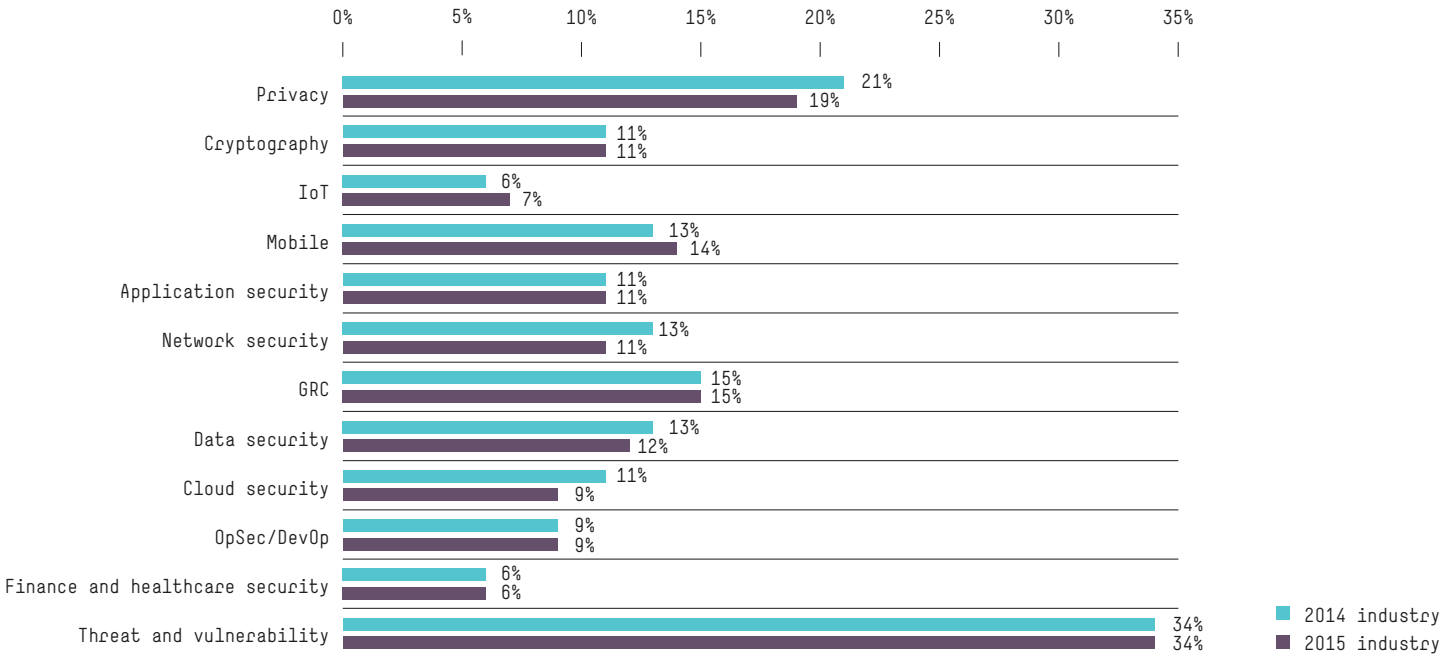


Figure 78 Presentation topics at industry-focused security conferences, 2014-15

For a lively privacy conversation, apparently academic conferences are the place to be. The academic conference sphere, we noted, tends to be a little more dynamic overall than that of industry. Here we see a significant rise in privacy-related talks—up 11% year over year. Threat and vulnerability issues, data security, and cryptography rounded out the year's biggest conference attractions. (What is not knowable from this research, but what may be visible as more data becomes available, is whether increases in one sphere are echoed in the other.) Several themes showed a strong increase in year-over-year presence including privacy, crypto, data security, IoT, and cloud (the latter two with a 50% increase year over year), while network security, GRC, and mobile declined.

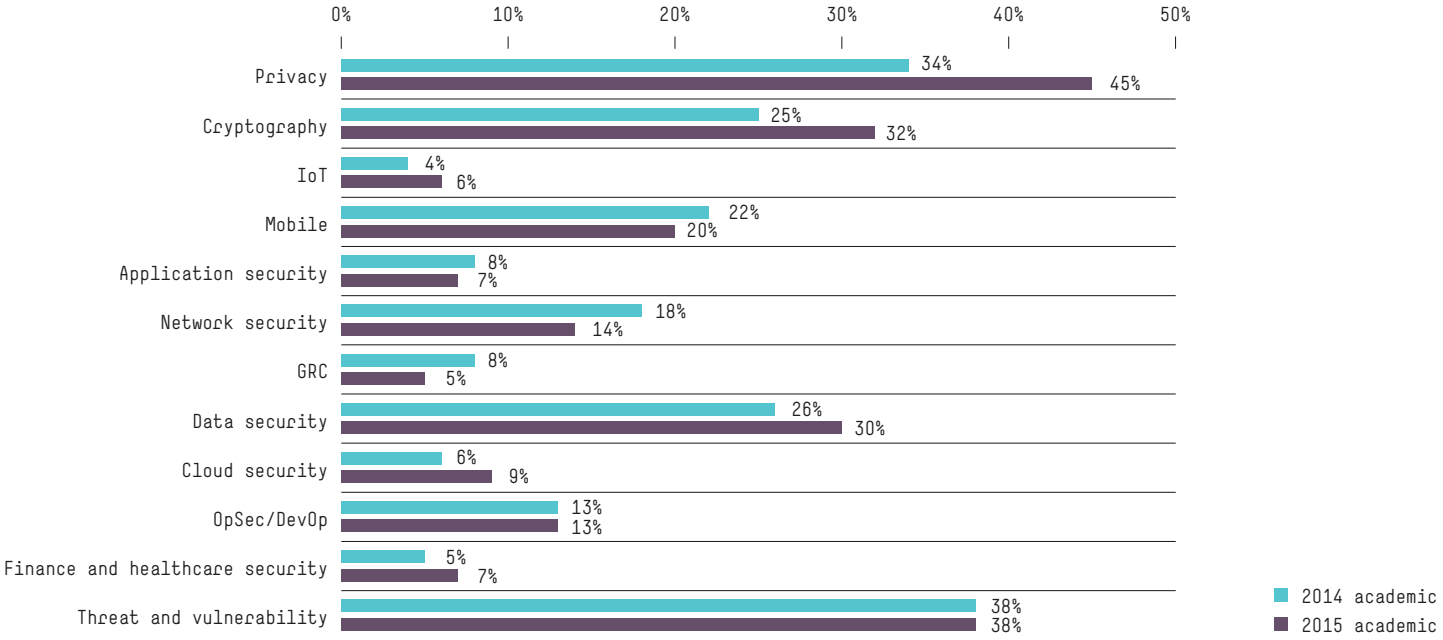


Figure 79 Presentation topics at academic security conferences, 2014-15

Industry	Change	Academic	Change
Software, system	0.80%	Application, analysis	4.40%
Application, system	0.70%	Network application	3.50%
System, exploit	0.70%	System, attack	3.40%
Mobile, data	0.70%	Application, attack	3.30%
Attack, Internet	0.70%	Software, attack	3.20%
Internet, device	0.70%	Data, detection	3.20%
Attack, detection	-0.50%	Application, device	-0.50%
Data, vulnerability	-0.50%	Application, protection	-0.50%
Data, analysis	-0.50%	Exploit, detection	-0.60%
Vulnerability, research	-0.70%	Mobile, application	-1.30%
Attack, threat	-0.80%	Mobile, device	-1.50%
Data, attack	-0.90%	Attack, device	-3.10%

Figure 80 Trending terms in conference abstracts (numbers indicate the percentage changes in popularity between 2014 and 2015)

Gram analysis

Finally, we analyzed the combinations of grams in our dataset to see which topics showed the most movement. Figure 80 shows, for industry and academic conferences, the 12 terms with the most and the least change in occurrence between 2014 and 2015. For the sake of readability we have used full words, but the data analysis is designed to use grams (essentially, word fragments to gather all appropriate tenses, combinations, singular/plural usages, and so on—for instance, "internet of things" in this chart actually covered "IoT," "Internet of Things," and "Internet-of-Things" in our data analysis.

As we see, terms related to threat intelligence and the Internet of Things increased in frequency of use in the industry sphere, while the academic side tended toward more diverse security-related terms (e.g., control flow, memory disclosure, key exchange, static analysis). The data also seems to bolster the traditional impression in both spheres that industry talks tend to focus more on the big security picture, while academic speakers delve into the nitty-gritty details of problems.

Fascinatingly, the industry chart appears to show a maturity process—specifically, the move from simpler presentations on preliminary analysis and protection to topics that are deeper and more intellectually weighty. In this light, it's easy to see the rise of terms associated with threat intelligence, best practices, security programs, and insider threats. Likewise, we see the diminution of terms related to incident response, web applications, application security, critical infrastructure, denial of service, and—ironically—big data. It is, in fact, entirely possible that we are watching data research transform into intelligence before our eyes.

Summary

If 2014 was dubbed the "Year of the Breach," it could be argued 2015 became the "Year of Collateral Damage." While Target and Home Depot previously grabbed headlines for the loss of customer data, the attacks on OPM and Ashley Madison demonstrated a level of impact beyond just credit card numbers. This year's Risk Report details the evolving nature of cybercrime as well as the developing legislation meant to curtail it. The report moves beyond the various techniques used by attackers, still driven primarily by financial interests, to delve into what defenders now face as they look to secure their enterprise.

2015 marks two decades of bug-bounty programs as well as the 10th anniversary of the ZDI program. During that time, various markets developed for researchers to highlight their work. The vulnerability white market has had a tremendous positive effect in securing the landscape by bringing researchers and vendors together. We expect the vulnerability market will continue to evolve as more and more vendors announce their own programs to incentivize research, further monetizing the value of independent security research.

We also anticipate regulations and legislation to affect the nature of disclosure. The impact of the Wassenaar Arrangement continues to have a ripple effect on the security research community. The recent inclusion of "intrusion software" under the Wassenaar Arrangement seems to be a backlash reaction to offensive security offerings. As the number of cyber-attacks continues to grow, there will likely be a corresponding response by governments to implement laws on how the information security industry operates. The end result of additional legislation related to security research will be that creating better protection solutions becomes harder and takes more time. This, in turn, increases the likelihood of successful breaches as the environment favors those researchers and agencies operating in the black market.

These legislative changes will certainly have an effect on the nature of disclosure. While the environment in which the information security community operates evolves, it is in all of our best interest to continue to find and disclose security bugs in popular software so vendors can fix things in a timely manner. The increasing complexity aside, it continues to be an endeavor worth doing.

This year also saw renewed efforts to decouple security and privacy. For enterprises, international data-privacy issues years in the making came to a head when Europe's highest court struck down the pact that allowed US and European interests to share data that has privacy considerations. The dissolution of the long-standing EU-US Safe Harbor agreement sent vendors to put together alternate data-transfer mechanisms even as regulators came knocking.

Continued world instability also brought the topic of surveillance and encryption into the minds of many. If surveillance manages time and again to seem like a white knight after terrorist incidents, encryption is often the dragon. In the days after the terrorist attacks on Paris, various simmering encryption-related debates were back on the boil, despite early evidence that encryption played no role in the terrorists' planning. Governments wish to monitor communications for significant threats, but doing so in a manner that does not interfere with civil liberties has proven problematic. This is coupled with the fact that current surveillance programs have not yielded the expected results.

While the breaches at OPM and Ashley Madison seem unrelated on the surface, both breaches had potentially terrible effects on people who never had direct contact with either agency. Despite the three years of credit counseling offered to persons whose names were revealed in the OPM breach, it's a relatively good bet that the stolen data wasn't meant for the hands of criminal gangs or identity thieves. Among the rich trove of data taken was, it is believed, data entered into Standard Form-86s, a document required for the background checks needed to obtain a security clearance. The form provides a great deal of information about one's family, friends, and associates—for security and intelligence professionals, a delicate situation. In other words, the true targets of the breach may be people who never consented to inclusion in the OPM database.

In the case of the Ashley Madison breach, information about an individual could potentially be derived even if it did not specifically appear in the data (e.g., a spouse's name and address would be obvious to a nosy neighbor). Again, even if it is unlikely the data leaked will end up being used by identity thieves, it could certainly have life-changing consequences. It's chilling to think that the exposure of data accessible through the Internet could have such a life-altering effect, but as more and more data migrates online, the scenario is likely to repeat itself unless data protections—namely privacy safeguards—are held firmly in place.

In the realm of security updates, the record number of point fixes for individual issues shows vendors are capable of keeping up with the current rate of vulnerability disclosures. What is not clear is whether this rate is sustainable. As evidenced with Microsoft web browsers, the inclusion of wide-reaching defensive strategies demonstrates how these fixes disrupt classes of attacks in an asymmetric fashion. Instead of releasing patches to fix many different vulnerabilities, these defensive measures take out the entire class—at least for some period of time. Other vendors would do well to consider implementing similar strategies to disrupt classes of attacks.

Despite the advancement of defensive strategies, malware continues to be a pervasive piece of life online. However, this past year did see a shift in the focus of malware. While always disruptive, today's malware has become focused more on money than on disruption of services. The ever-present ATM has become the focus for many of these attacks, with malware authors targeting the users of ATMs and the machines themselves. While coordinated law enforcement efforts achieved takedowns of banking Trojan infrastructure, statistics show the attackers are capable of restoring services to the botnets in a surprisingly rapid fashion. As more and more of our financial transactions occur online, criminals will continue to target these transactions for profit. Put simply, if there is money to be made, there is money to be stolen. The industry must focus on securing these transactions to deprive attackers of the illicit income they so desire.

Our yearly analysis of trends in application security provides a unique snapshot of the state of applications security during the past year. As in previous years, all identified issues were classified according to the HPE Software Security Taxonomy. During 2015, the taxonomy extended further to include other assessment techniques and HPE Security Fortify products such as HPE Security Fortify on Demand (FOD).

Generally, the breakdown of application security issues between 2014 and 2015 is strikingly similar. Year-to-year changes in the rankings for the three kingdoms with the lowest representation (API Abuse, Code Quality, and Time and State) are primarily due to changes to the HPE Software Security Taxonomy itself; the most prevalent vulnerabilities remain the same for both years. Mobile applications present different issues from those seen in non-mobile applications. Security Features continues to be the most represented kingdom for both web applications and mobile applications. Still, mobile applications tend to see over 10% more issues related to security features than do web applications. For mobile applications, it's internal system information leaks that lead the most common list. Remediation of these mobile issues remains a concern. Only 48% of the mobile issues in our sample seem to have been remediated—a stark difference from the 92% we saw on the applications side. Their very large presence atop the list of most frequently encountered mobile vulnerabilities indicates that a substantial majority of the applications we saw are storing sensitive information on devices that can be left on restaurant tables, stolen from backpacks, and dropped in toilets.

In security operations, the reactive nature of security monitoring is commonly the subject of complaints. In a reactive system, events must occur before they can be detected, as opposed to the more proactive prevention approach. The reactive-proactive conversation must occur within the context of the technology available, the most common of which is SIEM. While the use of security intelligence systems has been shown to equate to potentially millions of dollars in savings, implementation is not without its hazards. An operations analyst's ability to detect an event is predicated on his ability to see relevant event data. As data sources continue to grow, enterprises will need the capability of storing and analyzing the multitude of events gathered by various sensors. This requires investments in both people and technologies. While these investments have an initial outpouring of capital, the savings will be seen in preventing and responding to the inevitable breach.

In the coming years, the complexities of legislation and international events will have a greater impact in the realms of security and privacy. As a result, network defenders need to understand the complexities of privacy issues as thoroughly as they understand the impact of security vulnerabilities. Instead of symmetric responses to threats, tomorrow's network defender must understand how to respond asymmetrically to threats through automated analysis, wide-reaching fixes, and a community-based defense. While the threat of cyber-attack is unlikely to go away, thoughtful planning can continue to increase both the physical and intellectual price an attacker must pay to successfully exploit an enterprise.

Authors and contributors

The HPE Cyber Risk Report is an annual collaboration.

Authors

Brandie Anderson

Sue Barsamian

Dustin Childs

Jason Ding

Joy Marie Forsythe

Brian Gorenc

Angela Gunn

Alexander Hoole

Howard Miller

Sasi Siddharth Muthurajan

Yekaterina Tsipenyuk O'Neil

John Park

Oleg Petrovsky

Barak Raz

Nidhi Shah

Vanja Svajcer

Ken Tietjen

Jewel Timpe

Contributors

Matt Gibbs

Michele Huresky

Alvaro Muñoz

Joe Sechman

Peter Szabo

Daniel Trauner

ReversingLabs

Sonatype Inc.

Glossary

Amicus brief

A brief filed to a court by someone who is not a party to a case on which they are commenting (amicus curiae, "friend of the court").

API (application programming interface)

A set of tools and resources that provide various functions developers can utilize when creating software.

Ashley Madison

Ashley Madison is a Canada-based online dating service and social networking service marketed to people who are married or in a committed relationship. The company was breached in July 2015.

ASLR (address space layout randomization)

A security mechanism where the locations of important elements of a program in memory are randomized in order to make them harder for an attacker to find and utilize. This increases the difficulty for the attacker to perform particular types of exploits that rely on jumping to particular address areas of memory.

ATM (automated teller machine)

An electronic telecommunications device that enables the customers of a financial institution to perform financial transactions, particularly cash withdrawal, without the need for a human cashier, clerk, or bank teller.

Buffer overrun/overflow

A buffer overflow is a type of vulnerability that arises when a program writes an excessive amount of data to the buffer, exceeding the capacity of the buffer and then overwriting adjacent memory. This type of vulnerability may be exploited to crash programs or, with the correct manipulation by a skilled attacker, used to execute arbitrary code on a targeted computer. Buffer vulnerabilities can be avoided by the use of bounds checking, which checks the capacity for inputs before they are written.

C&C: See Command and control

CEN (European Committee for Standardization)

An EU body charged with establishing standards for goods originating from any of the Union's 28 member countries.

Circuit Court (US)

In the US, the federal system of appellate courts above the District Court level and below the US Supreme Court. There are 12 geographically defined circuits, including one for Washington, DC. In addition, there is an additional United States Court of Appeals for the Federal Circuit whose (nationwide) jurisdiction is based on subject matter.

Cookiejacking

Cookiejacking is a form of hacking wherein the attacker can gain access to session cookies of a browser's user.

Command and control (C&C)

As with many terms used in computer security, this term has been borrowed from the military. Similar to the military use of the term it means a method of exercising authority over resources; for example, a commanding officer commanding his troops. This term is often used in the context of malware and botnets in particular, where a structure is set up to command and control many compromised computers from either a centralized, or in some cases, decentralized position. A centralized command and control structure might be a single server that compromised computers connect to in order to receive commands. A decentralized command and control structure could be one in which compromised computers connect to a peer-to-peer network, where commands are spread through the network from many possible nodes. Command and control is also known as C2.

Command injection

Command injection occurs when an attacker is able to pass unsafe data to a system shell via a vulnerable application so that the unsafe data is then executed on the targeted system. The result therefore of a successful command injection attack is the execution of arbitrary attacker-supplied code on a targeted system. The risk of command injection attacks can be mitigated by appropriate input checking and validation.

Cross-frame scripting

A form of cross-site scripting attack, in which attackers exploit a vulnerability in a web browser in order to load malicious third-party content that they control in the frame of a webpage on another site. This attack may allow an attacker to steal sensitive information, such as login details, that may be input into the frame because the targeted user believes the request for login details came from the legitimate site.

Cross-site scripting

An attack that occurs when an attacker exploits a vulnerability in web applications in order to inject malicious code into client-side code that is delivered from a compromised website to an unsuspecting user. The code that is delivered to the user is trusted, and hence executed, as it appears to come from a legitimate source. These types of attack occur due to insufficient checking and validation of user-supplier input. Attackers may use this type of attack in order to bypass access controls or steal sensitive data.

Glossary

CVSS (Common Vulnerabilities Scoring System)

The Common Vulnerability Scoring System (CVSS) is an open framework for communicating the characteristics and severity of software vulnerabilities. CVSS consists of three metric groups: Base, Temporal, and Environmental. The Base group represents the intrinsic qualities of a vulnerability, the Temporal group reflects the characteristics of a vulnerability that change over time, and the Environmental group represents the characteristics of a vulnerability that are unique to a user's environment. The Base metrics produce a score ranging from 0 to 10, which can then be modified by scoring the Temporal and Environmental metrics. A CVSS score is also represented as a vector string, a compressed textual representation of the values used to derive the score.

DEP (data execution prevention)

A security measure used by modern operating systems that is intended to prevent the running of malicious code on an affected system. It operates by marking areas of memory as either executable or non-executable and raises exceptions when code attempts to run from areas that are deemed non-executable.

DLL (dynamic link library)

A dynamic link library (DLL) is a collection of small programs, any of which can be called when needed by a larger program. DLLs are Microsoft's iteration of the "shared library" concept used on other platforms.

Exploit

Code written expressly to take advantage of the security gap created by a particular vulnerability in order to deliver a malicious payload. Exploits may be targeted at specific organizations or used en masse in order to compromise as many hosts as possible. Delivery mechanisms utilize many different technologies and vehicles and often contain a social engineering element—effectively an exploit against vulnerabilities in human nature in order to make the victim take a particular action of the attacker's choosing.

External leakage

An external information leak occurs when system data or debugging information leaves the program open to a remote machine via a socket or network connection.

National Security Letters

A subpoena letter issued by the US federal government in order to gather information related to issues of national security. The Patriot Act gave the FBI greatly expanded power to demand certain records, including ISP information.

OPM (Office of Personnel Management)

In the US, the Office of Personnel Management is a federal department that handles human resources issues for government employees.

PCI-DSS

The Payment Card Industry Data Security Standard was developed by the payment card industry as a framework for secure handling and storage processes for information related to credit, debit, and similar cards. It is managed by the PCI Security Standards Council.

PII

Personally identifiable information. The definition of what kinds of data are to be treated as PII varies from jurisdiction to jurisdiction.

POODLE (Padding Oracle On Downgraded Legacy Encryption)

The POODLE attack was a 2014 man-in-the-middle exploit that took advantage of Internet and security software clients' fallback to SSL 3.0.

Remote code execution (RCE) vulnerability

A vulnerability that allows attackers to execute their own code on a target system. Depending on the vulnerability used, the RCE may be executed with either user- or system-level permissions.

ROP (return oriented programming)

An exploit technique that allows an attacker to execute code while bypassing certain types of defense-in-depth measures, such as ASLR.

Safe Harbor

Generally, a provision within a statute or regulation that states that specified behaviors are not in violation of that law. In the privacy realm, it refers to a framework developed by the US and the European Union describing how US companies could receive, handle, and use European citizens' personally identifiable information (PII) without running afoul of European privacy laws.

Secure Data Act

Proposed US legislation that would forbid federal agencies from requiring private enterprises to build technology into products for government surveillance purposes.

Shellcode

A small piece of code used as the payload during the exploitation of a vulnerability. While these types of payloads typically start from a command shell, any code that performs a similar function is generically referred to as shellcode.

Small and midsize business (SMB)

A small and midsize business (SMB) is a business which, due to its size, has different IT requirements—and often faces different IT challenges—than do large enterprises, and whose IT resources (usually budget and staff) are often highly constrained.

STIX (Structured Threat Information Expression) and TAXII (Trusted Automated Exchange of Indicator Information)

An open community-driven effort and a set of free, available specifications that help with the automated exchange of cyber-threat information. This allows cyber-threat information to be represented in a standardized format. They are not pieces of software themselves, but rather standards that software can use. The combination of STIX and TAXII allows participants to more easily share threat information with constituents and peers.

Glossary

SOC (security operations center)

A business unit that deals with enterprise security issues, both ongoing and responsive, including the processing of data, alerts, and logs pertaining to the enterprise's security. Does not necessarily refer to a physical space.

Trojan

Malicious software that, unlike worms or viruses, is unable to spread of its own accord. There are many different types of Trojans that are used in conjunction with other types of malware in order to perpetrate computer crime. One of the most notorious types is a remote access Trojan (RAT) that can be used by a remote attacker to access and control a victim's computer.

USA Freedom Act

A 2015 law that restored certain provisions of the Patriot Act that had sunsetted earlier in the year. The name is an acronym for "Uniting and Strengthening America by Fulfilling Rights and Ensuing Effective Discipline Over Monitoring Act."

USA Patriot Act

In the US, a set of laws passed in the wake of the September 11, 2001, terrorist attacks. The name is an acronym for "Uniting and Strengthening America by Providing Appropriate Tools Required to Intercept and Obstruct Terrorism Act."

Use-after-free

A use-after-free vulnerability can occur when memory is allocated to an object that is used after it is deleted (or deallocated). Good programming practice dictates that any reference pointing to an object should be modified when the memory is deallocated, to keep the pointer from continuing to make the area of memory where the object once resided available for use. (A pointer in this abandoned condition is broadly called a "dangling pointer.") If the pointer isn't modified and tries to access that area of memory, the system can become unstable or corrupt. Attackers can use a dereferenced pointer in a variety of ways, including execution of malicious code.

Vulnerability

Defects or bugs that allow for external influence on the availability, reliability, confidentiality, or integrity of software or hardware. Vulnerabilities can be exploited to subvert the original function of the targeted technology.

Wassenaar Arrangement

Agreement to establish the Wassenaar Arrangement was reached on 19 December 1995 in Wassenaar, near The Hague, in the Netherlands. The Wassenaar Arrangement has been established in order to contribute to regional and international security and stability, by promoting transparency and greater responsibility in transfers of conventional arms and dual-use goods and technologies, thus preventing destabilizing accumulations. The aim is also to prevent the acquisition of these items by terrorists. There are currently 41 participating states (countries).

Worm

A self-contained malicious program that is able to spread of its own accord. The classification "worm" is only used to describe the ability to spread without a host file (as may be the case with computer viruses) and worms contain many different and varied payloads beyond spreading from host system to host system.

YARA

YARA is a tool to aid malware researchers in identifying and classifying malware samples. YARA allows for the creation of malware family descriptions based on textual or binary patterns.

Zero day

A previously unknown vulnerability for which no patch from the vendor currently exists. It is referred to as a zero day because the vendor has had zero days to fix the issue.

Learn more at
hp.com/go/hpsr

Sign up for updates

★ Rate this document

Hewlett Packard Enterprise

4AA6-3786ENW, February 2016