KENNA Security

Cyentia 119 INSTITUTE

PRIORITIZATION TO PREDICTION

Volume 6: The Attacker-Defender Divide

https://t.me/learningnets

KENNA Security

PRIORITIZATION TO PREDICTION

VOLUME 6: THE ATTACKER-DEFENDER DIVIDE

This research was commissioned by Kenna Security. Kenna collected and provided the dataset to the Cyentia Institute for independent analysis for this report.

Kenna Security is the enterprise leader in risk-based vulnerability management. Kenna Security solutions enable organizations to work cross-functionally to determine and remediate cyber risks. They leverage machine learning and data science to track and predict real-world exploitations, empowering security teams to focus on what matters most. Headquartered in San Francisco, Kenna serves nearly every major vertical and counts CVS, KPMG, and many Fortune 100 companies among its customers.

Find out more at www.kennasecurity.com.

Cyentia INSTITUTE

Analysis for this report was provided by the Cyentia Institute. Cyentia is a research and data science firm working to advance cybersecurity knowledge and practice. We do this by partnering with security vendors and other organizations to publish a range of high-quality, data-driven content like this study.

Find out more at www.cyentia.com.

https://t.me/learningnets

Overview & Key Findings

(1) Every object moves in a straight line unless acted upon by a force.
(2) The acceleration of an object is directly proportional to the net force exerted and inversely proportional to the object's mass.
(3) For every action, there is an equal and opposite reaction.

–Newton's Laws of Motion

Newton first proposed these laws in his Philosophiae Naturalis Principia Mathematica in 1687. It's widely considered one of history's intellectual masterpieces and helped fuel the Scientific Revolution. Newton obviously wasn't thinking of cybersecurity when developing these laws, but our latest scientific investigation into the mechanics of vulnerability remediation and exploitation find them very applicable.

Cybersecurity is often described as a game of cat-and-mouse, in which the opposing forces of attackers and defenders continually vie for advantage. Vulnerability management (VM) is a microcosm of that struggle. Vulnerabilities are discovered and disclosed, patches are created, exploit code drops, defenders begin remediation, attackers begin exploitation, and the cycle goes on. In this chaotic environment, it's very difficult to sustain positive momentum toward reducing risk exposure to the organization. Attackers use that to their advantage, widening the attacker-defender divide. But it doesn't have to be that way.

The Prioritization to Prediction (P2P) report series, a collaboration between Kenna Security and the Cyentia Institute, seeks to better understand the science of vulnerability management. In this sixth volume, we identify and measure the forces acting upon vulnerability exploitation and remediation, offering valuable insights needed to leverage them to your advantage.

Key Findings

Based on multiple independent data sources, we observed evidence of exploitation in the wild against 473 of the more than 18,000 CVEs published in 2019. The entire focus of this study is learning everything we can about those 473 CVEs. For example...

A mere 6% of those 473 vulnerabilities ever reached widespread exploitation by more than 1/100 organizations. The fact that an exploit is "in the wild" does not mean it's raging hog wild across the internet.

There is no "typical" vulnerability lifecycle. Only 16% of the CVEs we studied followed the most common sequence of Reserved-Patched-Scanned-Published-Exploited. Several key milestones tend to converge surrounding CVE publication.

Exploit code was already available for >50% of vulnerabilities (eventually exploited in the wild) by the time they published to the CVE List. Thankfully for defenders, patch releases coincide with publication for over 80% of those CVEs.

It's common to use a 30-day window for patching vulnerabilities. Those following that guidance might like to know that initial exploitation in the wild was detected within that 30-day window for about half the vulnerabilities we studied.

It generally takes defenders a month after a patch is released to remediate 50% of the vulnerable assets in their environment. By comparison, attackers reach 50% of max exploitation prevalence across target organizations in about 2.5 months.

That last fact makes it seem like defenders own the momentum. But over the ~15 month period we studied, attackers had the upper hand for about 9 months compared to 6 months for defenders.

What's more, the period of defender momentum is drastically reduced when an exploit is released before the patch becomes available (attackers gain the advantage for 12 out of 15 months).

https://t.me/learningnets

Enumerating Exploited Vulnerabilities

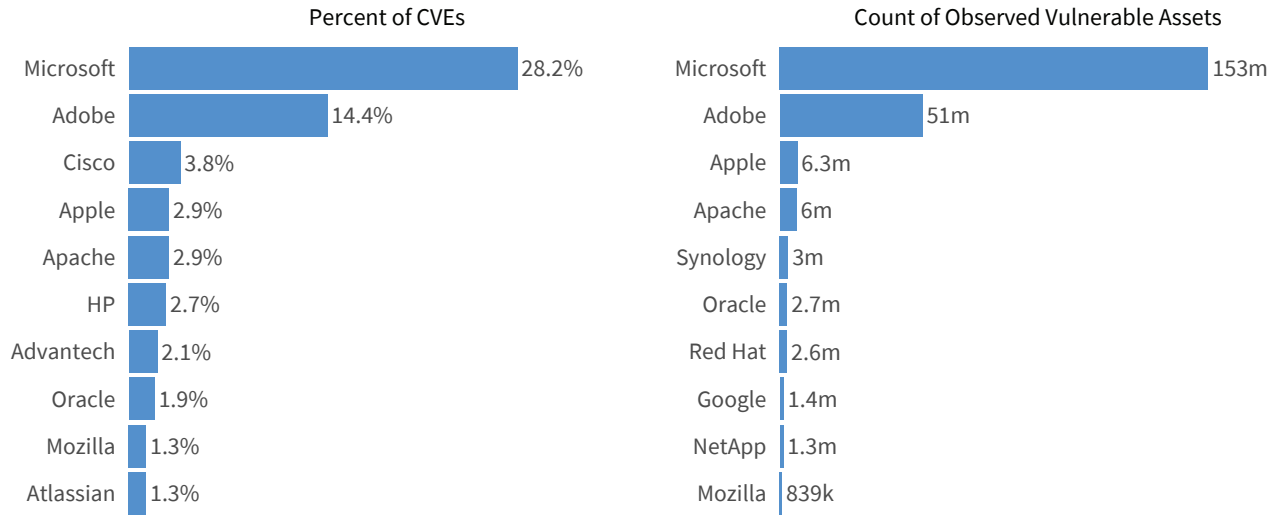
For this edition of the Prioritization to Prediction series, we want to more fully investigate the timeline of exploitation and compare it to that of remediation. That involves examining the entire lifecycle of a vulnerability, from first discovery through the spread of exploitation across the internet to when it eventually (hopefully) becomes widely-addressed. We focus on vulnerabilities published to the CVE List in 2019 and track their progress through the first half of 2020. We chose this timeframe for two reasons: First, we think more recent vulnerabilities represent the path of future vulnerabilities better than say, a flaw from 2001. Second, we have solid data sources for exploitation of these vulnerabilities in the wild and their remediation across hundreds of organizations during this time period.

We expect exploitation "in the wild" may raise questions from some readers. So, let's be clear about what we're using in this study. We received exploitation data from a total of six independent sources. Five of these are compiled by Kenna Security, including two intrusion detection services, two open source threat intel services, and one malware analysis service.

We also include data generously shared by Fortinet, the largest provider of information on exploited CVEs. Fortinet has one of the largest security device footprints in the industry,[1] granting wide visibility into exploit detection. They continue to support exploitation research like the Exploit Prediction Scoring System (EPSS). It might sound cliche´, but it's true that sharing is a sincere form of caring about the security community. Our thanks to Fortinet and others who contribute data that benefits us all.

Using these combined sources we identified 473 unique 2019 CVEs with evidence of exploitation in the wild. We start our analysis by asking which products these vulnerabilities affect and how often they show up in real-world environments. Figure 1 tackles that question by listing the most-affected vendors as a percentage of these 473 CVEs (left) and as a total count of vulnerable assets detected by organizations (right). As we've seen in previous volumes, Microsoft leads by both measures and the two lists overlap substantially. Beyond the top 10 shown here, these vulnerabilities affect another 106 unique vendors.

Figure 1: Breakdown of vendors associated with the 473 exploited CVEs studied in this report



[1] *Source: IDC Worldwide Security Appliance Tracker, April 2020 (based on annual unit shipments of Firewall, UTM, and VPN appliances)*

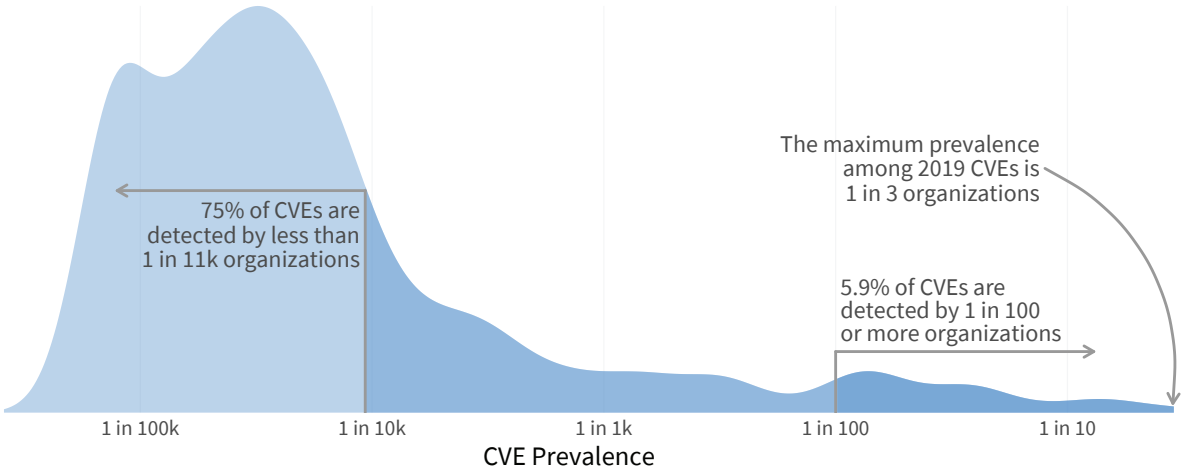
https://t.me/learningnets

Next, we measure the prevalence of exploitation activity targeting these vulnerabilities across organizations. For this, we analyze detected exploitation in the wild within a six-month window following publication of each CVE. Why six months? We want to ensure that each vulnerability has the same amount of exposure time for measuring exploitation activity. The last CVE of 2019 in our dataset was published on December 30, 2019 and our exploitation data runs through June 30, 2020.[2]

The distribution of exploitation prevalence for CVEs in our sample can be seen in Figure 2. Three-quarters of these vulnerabilities exhibit very limited exploitation in the wild, having been detected by less than 1 in 11,000 organizations! Following the long tail out to the right, about 6% of CVEs were seen by one in a hundred organizations and only a small fraction hit the peak prevalence of one in three firms.

We suspect some may find these results rather surprising. Some treat exploit activity as one massive event whereby an organization's probability of exploitation goes from 0 to 1 overnight. But that's simply not the case. Only a small portion of vulnerabilities ever sees widespread exploitation, and that generally takes some time to progress from Patient Zero to peak prevalence (more on that later). This concept should be fairly intuitive for those living under the shadow of COVID-19 in 2020. Thankfully, not every virus is a super-spreader capable of infecting tens of millions of people worldwide.

Figure 2: Prevalence of exploitation in the wild targeting 2019 CVEs (based on percentage of organizations)



Note that these measures of prevalence rely on network sensors detecting exploitation activity. While attackers may be attempting to use an exploit targeting a specific technology, that technology may not actually exist in the target's environment. And that brings up an interesting question about how targets are selected for exploitation attempts. It seems logical that attackers would favor widely adopted products for exploitation, but the data reveals that to be a very weak correlation. We'd like to study that more closely at some point, but it's well beyond the scope of this report.

A quick word on non-CVE vulnerabilities: It's true that focusing on CVEs misses vulnerabilities that, for one reason or another, fall outside the CVE process. We've clearly stated that limitation from the very first volume in this series. But how much of a limitation is that, actually? We have data to help answer that question.

About 95% of all vulnerabilities observed by firms in our dataset have a CVE. Interestingly, it's much lower for network appliances and devices (73%). But the overall lesson is that CVEs offer good coverage of the vulnerabilities detected by the various scanners used by organizations around the world as a foundation of their VM programs.

[2] *In practice most organizations see an exploitation attempt within the first six months and our results do not change qualitatively if we allow for a larger window for those CVEs that could potentially have a wider window.*

https://t.me/learningnets

Revisiting the Vulnerability Lifecycle

Now that we know a bit more about the nature of these CVEs, let's discuss the vulnerability lifecycle. We did something like this way back in the very first Prioritization to Prediction report, but it bears review and updating to prepare for what's to come in this edition. In general, a vulnerability will pass through some combination of one or more of the following milestones during its lifecycle:

Creation: This appears obvious, but it is worth stating nonetheless. A vulnerability is created when flawed code is written and released in a vulnerable state. It has the potential for exploitation regardless of whether or not it has been discovered, disclosed, or developed into exploit code. We rarely have visibility into the date this occurs and do not analyze this milestone in this research.

Discovery: Bug hunting has a lifecycle of its own, the nuances of which we won't get into here. Suffice it to say that when someone learns that a vulnerability exists, it has been discovered.

Disclosure: When a vulnerability is discovered, it's generally reported to the vendor of the affected product (at least that's what the CVE guidelines recommend). Typically, the vendor will confirm the vulnerability's existence, and, in most cases, request a CVE ID for the vulnerability.

CVE Reserved: We take direction from MITRE for this one: "Once there is enough information to confirm the vulnerability exists and that it affects a covered product, the CVE Team [or CNA] will reply to the requester with a CVE ID. The CVE ID is considered 'RESERVED' at this stage. Descriptions with details of the vulnerability will only be added when the vulnerability is made public."[3]

CVE Published: A CVE goes from reserved to published when it's officially added to the CVE List. In particular: "The CVE List is a dictionary of publicly disclosed cybersecurity vulnerabilities and exposures that is free to search, use, and incorporate into products and services, per the terms of use."[4] The CVE List feeds the U.S. National Vulnerability Database (NVD).

Patch Release: One of the tenets of coordinated disclosure is to withhold full release of all information on a vulnerability until the vendor has sufficient time to address the issue and develop a patch. Patch availability is a critical enabler of remediation for defenders.

Vulnerability Detection: Vulnerability scanners need to know what to look for in order to detect vulnerabilities that affect enterprise assets. Implementation details are different among vendors, but most maintain some form of signatures. Naturally, these signatures take time to create and distribute, which introduces lag before vulnerabilities can be detected. We've chosen to focus on the first detection by a vulnerability scanner in a live environment as a proxy for this because Kenna has strong visibility into this milestone.

Remediation: This involves patching, fixing, or otherwise addressing a vulnerability such that it's no longer an exposure. Individual vulnerabilities on specific systems can be remediated, but what's really in view here is a process by which organizations remediate all relevant vulnerabilities across their environment. We've studied this process exhaustively in prior P2Ps, and we'll add some additional lessons in this one.

[3] *From https://cve.mitre.org/cve/researcher_reservation_guidelines*

[4] *From https://cve.mitre.org/cve/*

https://t.me/learningnets

Exploit Code: For some vulnerabilities, exploit code may be published in a variety of forms. Sometimes it's just source code with or without a payload. Sometimes it's a working proof of concept (PoC). Sometimes the code is integrated into an exploitation framework such as Metasploit. As we've found in previous P2Ps, vulnerabilities with exploit code are significantly more likely to be exploited in the wild.

Exploitation: Attacks targeting a vulnerability in the wild constitute exploitation. The success of those attacks is ultimately what vulnerability management programs work to prevent. The good news (if we can call it that) is that not all organizations are targeted at once. So, observing any exploitation and/or the spread and prevalence of exploitation may (and should) escalate remediation priority.

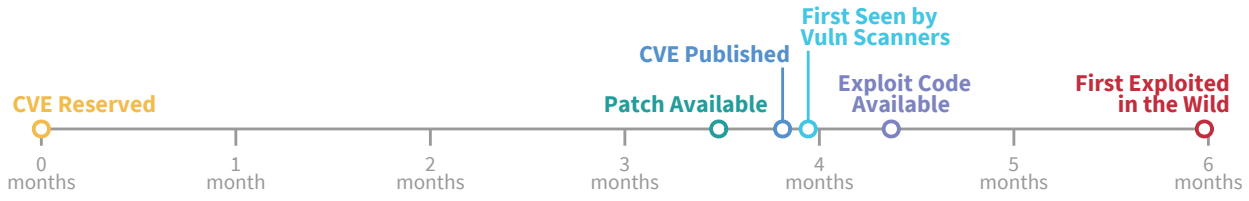
In order to analyze these 473 vulnerabilities, we had to make some concessions on which of these milestones to include. A few dates proved difficult to discern, so we made the hard choice of focusing on those which we could collect with reasonable effort and reliability. Often, the first date we observed and collected for vulnerabilities was when the CVE was reserved, so earlier events such as vulnerability creation and discovery won't be discussed much as we move forward into the next section.

Getting Things in Proper Order

Not all vulnerabilities have all the milestones from our list in the previous section, nor do the milestones always occur in that exact order. A proper understanding of this timeline is essential to several important risk-relevant questions such as: How quickly are vulnerabilities exploited? Which events act as a catalyst for exploitation? Can we remediate quickly enough to minimize risk exposure? In the words of Andy Weir's *The Martian*, "we're gonna have to science the sh*t out of this" to answer these questions. Let's start simple by establishing some basic principles about the order of things in the vulnerability lifecycle.

We stand by the statement that there is no "typical" sequence (we'll explain why in a minute). But if you insist on giving a single typical timeline—being the analytical people that we are—we'd feel compelled to come up with a sequence that best represents the data collected on these 473 vulnerabilities. Anticipating that insistence, we went ahead and came up with the sequence in Figure 3.

Figure 3: A "typical" timeline of key milestones in the vulnerability lifecycle (based on median days from CVE Reserved)



Careful—don't start quoting stats from Figure 3 yet. It shows the median days for milestones in the vulnerability lifecycle, but we're about to show there's A LOT of variation and what you see here doesn't happen that often.

We created Figure 3 by calculating the median number of days between the first event and each subsequent event. Several interesting facts about the vulnerability lifecycle emerge from this chart.

The first public record of a vulnerability's existence almost always occurs in the form of a CVE being reserved or assigned. That record often doesn't contain much (or in some cases, any) information, but it does plant a flag in the ground and allows references to start popping up on our radar. Typically, nothing much happens for 3+ months after that.

https://t.me/learningnets

Thankfully, the exploitation in the wild is usually the last milestone in the sequence of events. It's worth noting, however, that there are rare instances where this occurs even before a CVE is reserved (a true Zero Day vulnerability). We show the first time a sensor detected active exploitation here, but we'll examine the spread of exploit activity later in this report.

The milestones in between CVE reservation and exploitation (patch availability, CVE publication, first recording by vulnerability scanners, and exploit code availability) rank in the order shown. But, in reality, these four milestones occur in close proximity to one another and frequently swap relative positions (see Figure 5 further below). It's not out of line to treat these four as one joint milestone in which activities take place as quickly as possible to give defenders what they need to begin remediation.

In reality, the vulnerability lifecycle isn't nearly as simple as Figure 3 depicts. Below you'll find the top 10 sequences of milestones we observed across the 2019 CVEs. Only 16% of the CVEs we studied followed the most common (topmost) sequence in Figure 4. That by itself tells you a lot about the "typical" vulnerability lifecycle—namely that it doesn't exist! By the time we get down to the last pattern in this plot, we're talking about barely over 2% of vulnerabilities—and there are still over 100 unique milestone strings below that.

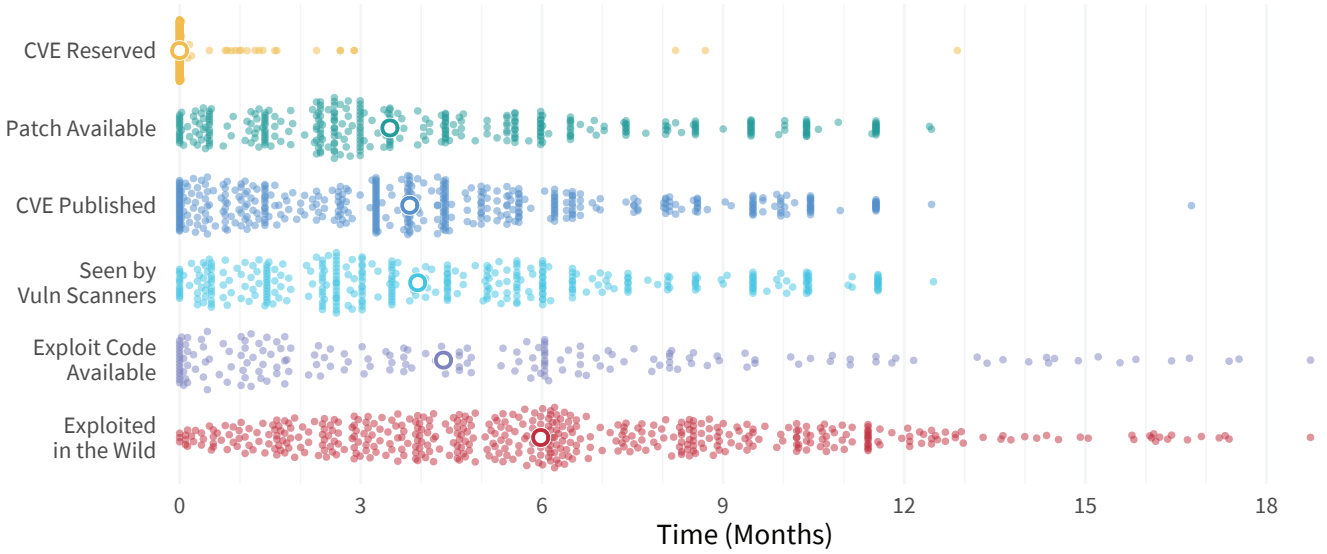
Figure 4: Top 10 sequences of key milestones in the vulnerability lifecycle

After seeing Figure 4, it should be clear that there's really no 'One String to Rule Them All' in the vulnerability lifecycle. Let's try another visual to help make sense of this chaotic process. Figure 5 measures the number of days each milestone occurred relative to when the CVE was reserved. Each dot marks the number of days-from-reserved for the milestones while the circles indicate where the median sits among those dots. From this, we glean the following insights:

- All milestones overlap—some almost completely. This reinforces our earlier comments about how those middle milestones in Figure 3 can occur in any order.
- It's not uncommon for those "middle" milestones to occur on or near the same day the CVE is reserved. When that happens, it indicates coordination among the various parties involved.
- The median for first exploitation in the wild falls a couple of months beyond all other milestones and the dots extend farther to the right. That's a statistical way of saying what we said earlier about exploitation generally occurring later.

https://t.me/learningnets

Figure 5: Distribution of dates associated with key milestones in the vulnerability lifecycle (days from CVE Reserved)

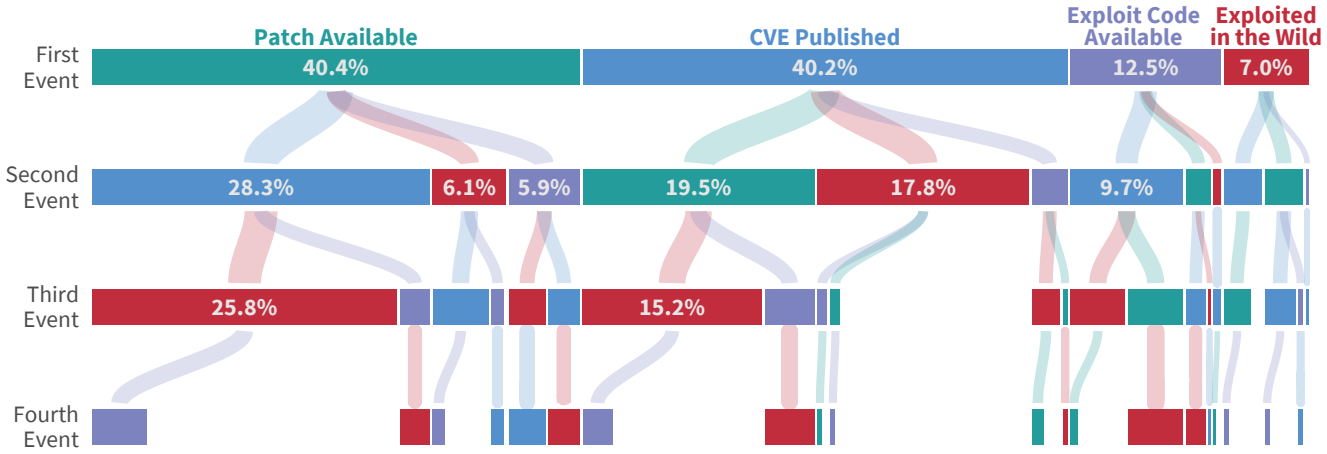


Ok, you might want to stand up and stretch before taking on this next chart. It's a bit of a doozy. But it's also accurate and, hopefully, helpful to our goal of Marie Kondo-ing things into the right order. To make this more manageable, we focus on four key milestones: CVE published, patch available, exploit code, and exploited in the wild.

The first row in Figure 6 shows how often each of those key milestones occurred first (excluding CVE reserved, which as discussed, typically is event zero). So, 40% of vulnerabilities started with a patch release, another 40% began with CVE publication, 12.5% went straight to exploit code, and 7% jumped right into exploitation in the wild.

Tracing the flows between the first and second event tiers breaks down what happened next. 28% of CVEs progressed from patch to CVE publication, 6% went from patch to exploitation in the wild, and another 6% saw exploit code released after the patch. Continuing down to the third event tier, 26% of CVEs followed the patch-CVE-exploitation path; 15% progressed from CVE to patch to exploitation in the wild. You get the picture. It just takes some work to see the whole thing.

Figure 6: Breakdown of all observed sequences of key milestones in the vulnerability lifecycle for 2019 CVEs

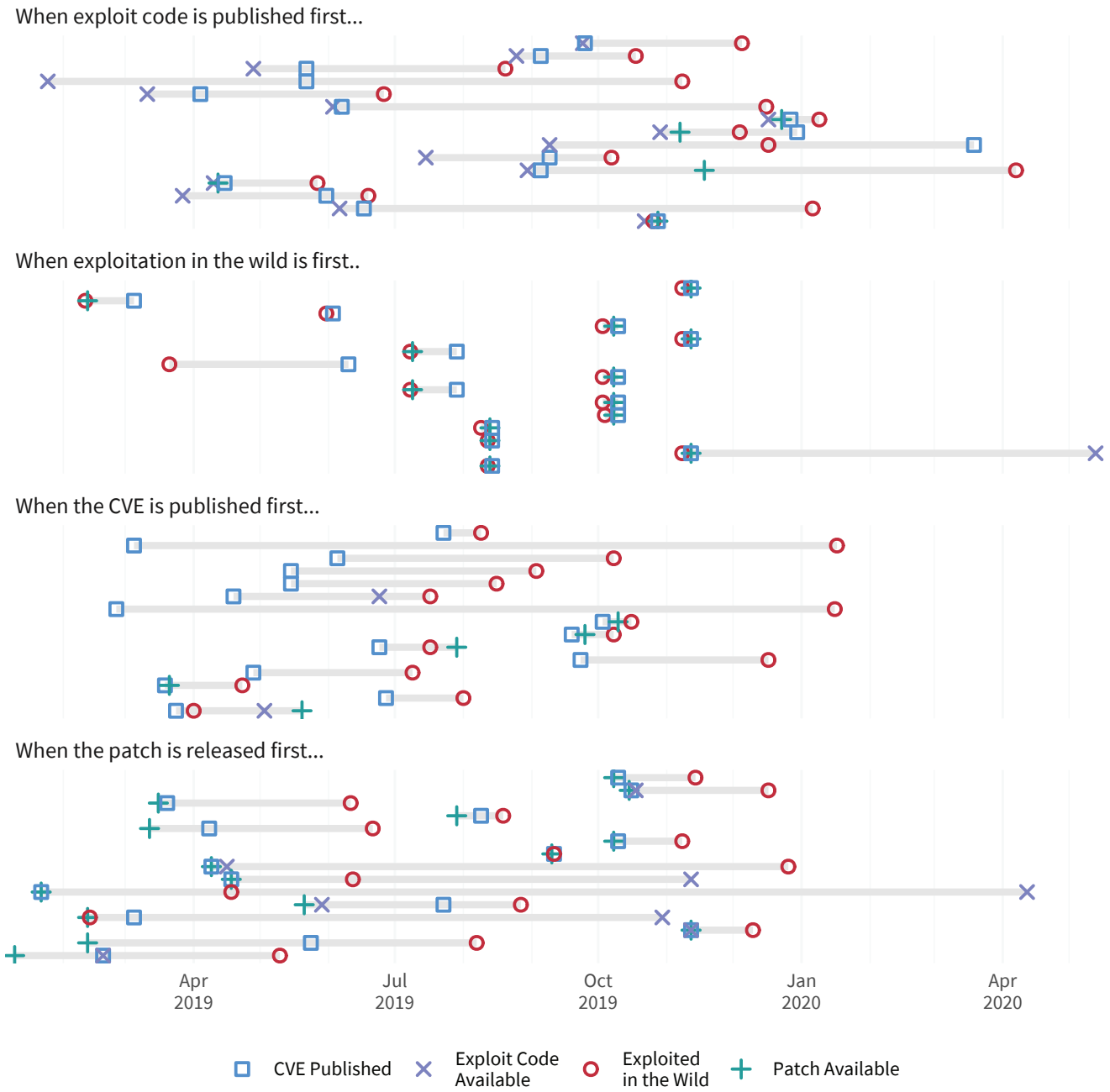


We've now developed a better understanding of how the vulnerability lifecycle plays out sequentially, but only scratched the surface on the timing of key events. Figure 7 scratches a layer deeper, and the next section gouges deep down to the heart of the matter.

https://t.me/learningnets

For each of the four milestones presented in Figure 6, to illustrate the lifecycle diversity among CVEs, we selected a sample of 15 vulnerabilities. Milestones associated with each vulnerability are plotted on the timeline in Figure 7 according to the date of occurrence. From this, it's apparent that the lifecycle of some vulnerabilities unfolds quickly and, for others, it's prolonged. The intervals between milestones range from balanced to highly irregular. Sometimes, events occur right on top of one another, and some are separated by a year or more. So, yeah...it's complicated.

Figure 7: Timelines of key milestones in the vulnerability lifecycle for a select sample of 2019 CVEs



"It's complicated" is honestly the main takeaway from this section. We've gone from a simplistic view that's rather misleading in Figure 3 to a complex view that's accurate yet mystifying in Figures 6 and 7. The best depiction to use will depend on your needs. If you want a hand-wavy way to quickly portray the vulnerability lifecycle, go with the former. To show what's really going on, however, you'll have to suss what you can out of the latter.

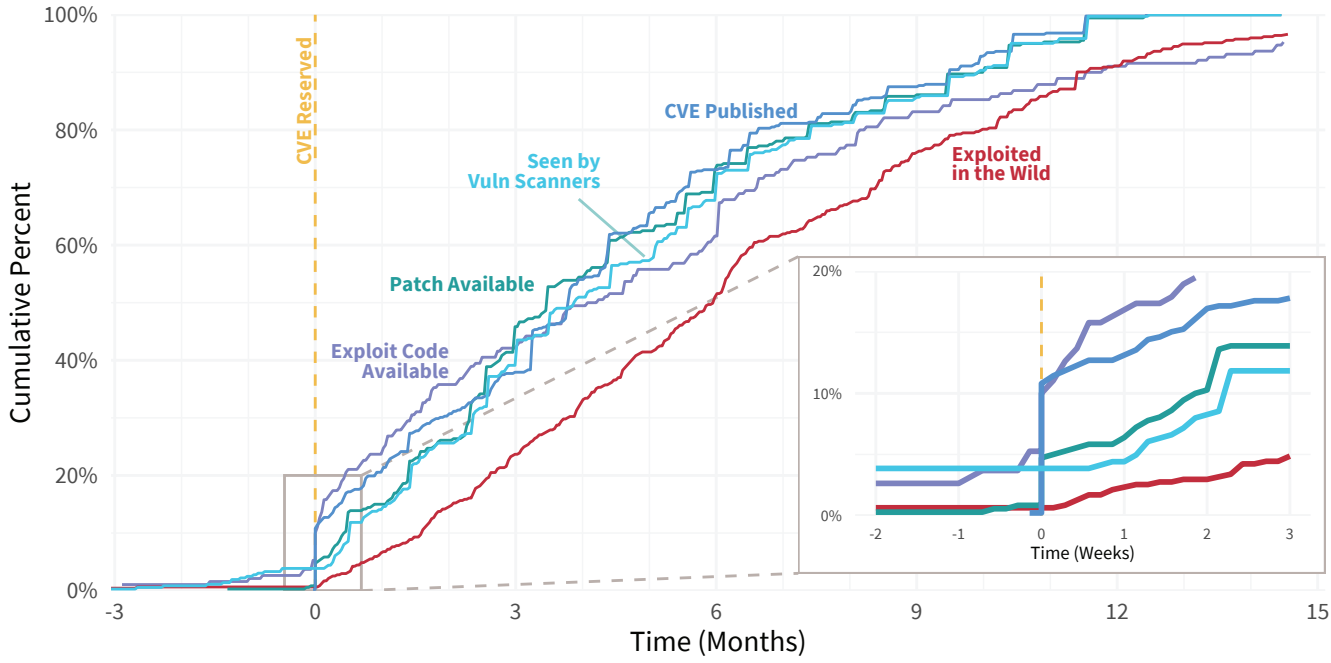
And guess what—putting on our best Billy Mays voice—we're not done yet! If you keep reading for the next 10 minutes, we'll throw in our patented, never-before-seen selection of time interval charts at no extra cost. Get 'em while supplies last!

https://t.me/learningnets

Timelines in the Vulnerability Lifecycle

This next section is going to be fun—at least for those who enjoy soaking in every last drop of insight from rich data visualizations. If that doesn't describe you, don't despair! We've liberally annotated to draw attention to the major takeaways from each chart. The idea here is to create a series of similarly formatted charts that measure the time intervals between key milestones in the vulnerability lifecycle. Beyond that brief explanation, the best way to get what we're trying to do is to jump right into the visualizations. So, let's do that.

Figure 8: Timing of several key milestones relative to when the CVE was reserved



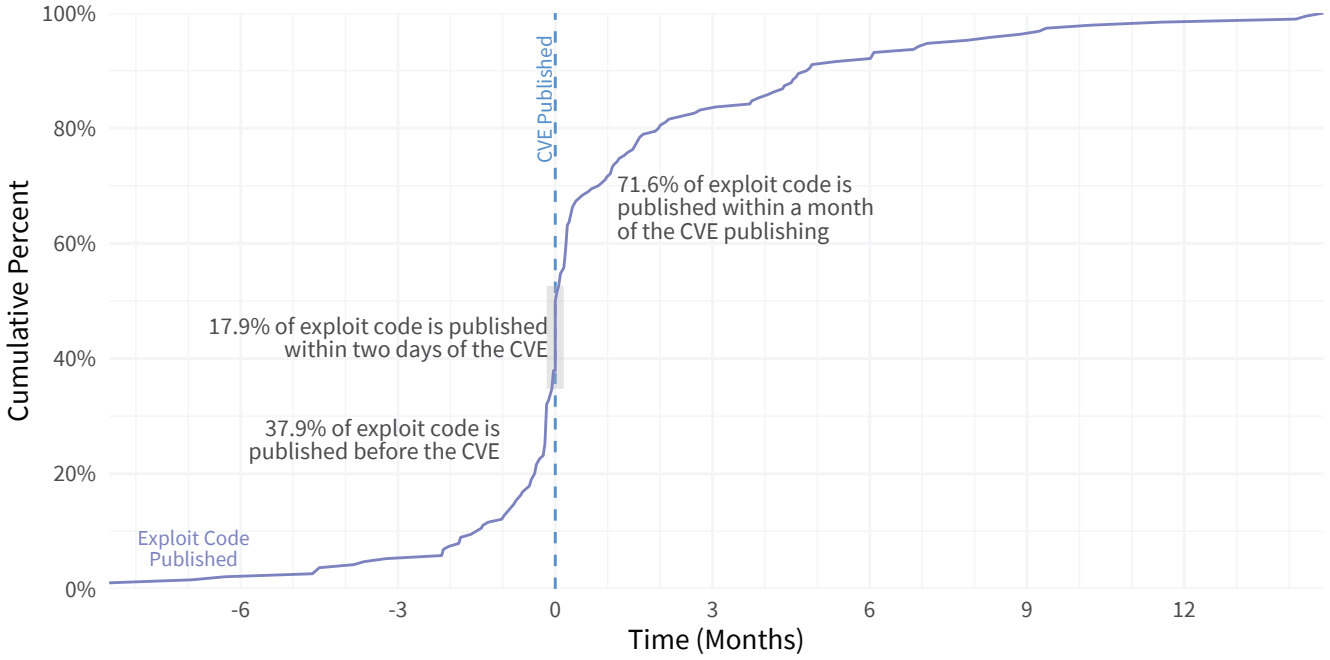
Importance: Reserving a CVE is generally the first public indicator of a vulnerability's existence. So, measuring the other milestones relative to the reservation date point makes intuitive sense.

Observations:

- From this, it's clear that sometimes CVEs take a page from Anthony Bourdain and require No Reservations. But the vast majority of activity occurs after reservation.
- If something does happen before the CVE is reserved, it's most likely the release of exploit code. Notice also how the line for exploit code availability shoots up on the day of CVE reservation. That's not ideal from a defensive standpoint because remediation can't begin before enabling milestones (patch, scanner signatures) emerge.

https://t.me/learningnets

Figure 9: Timing of exploit code release relative to CVE publication



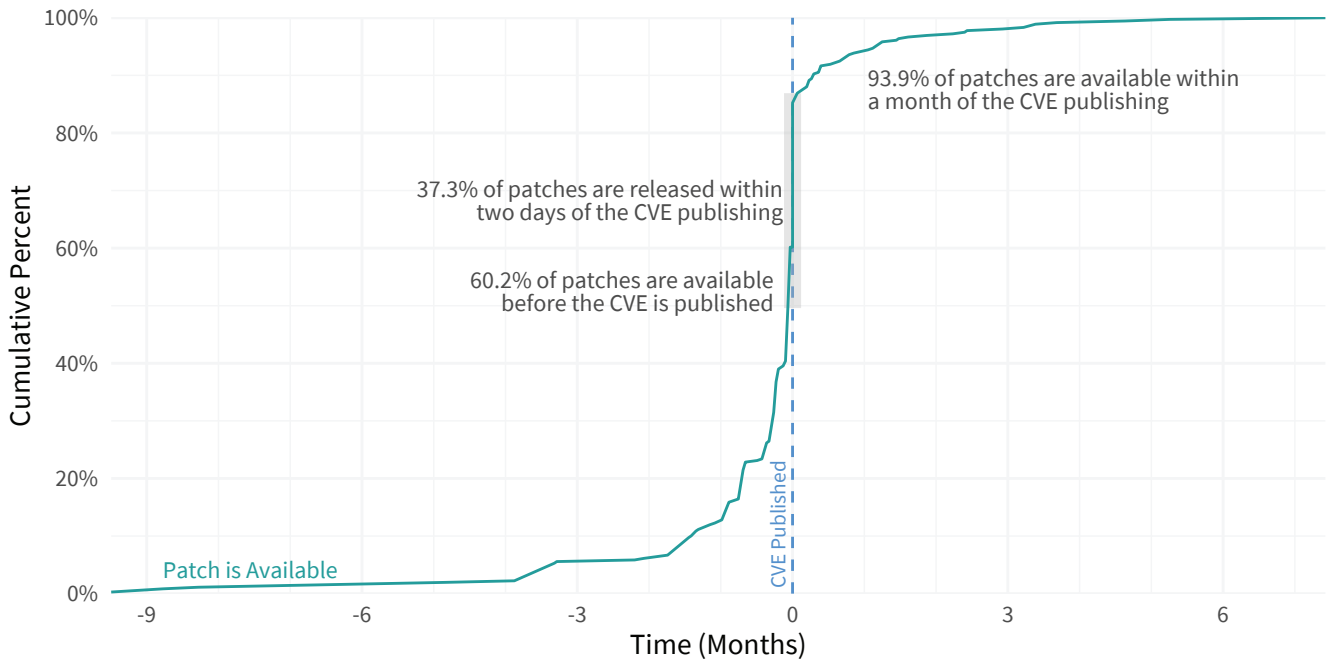
Importance: When a CVE is published, all the pertinent information becomes fully available to both attackers and defenders (though one or both groups often have it before publication). Knowing when exploits drop relative to the date of publication helps defenders understand how much time they have to mitigate exposure.

Observations:

- Within a day of publication, over 50% of vulnerabilities (known to be exploited in the wild) already have code available to exploit them. One month after a CVE is published, 75% have been weaponized.
- We'll soon show that the release of exploit code is highly correlated with active exploitation of vulnerabilities in the wild. Thus, defenders should treat a published CVE as a sign the clock is (and has been) ticking.

https://t.me/learningnets

Figure 10: Timing of patch availability relative to CVE publication



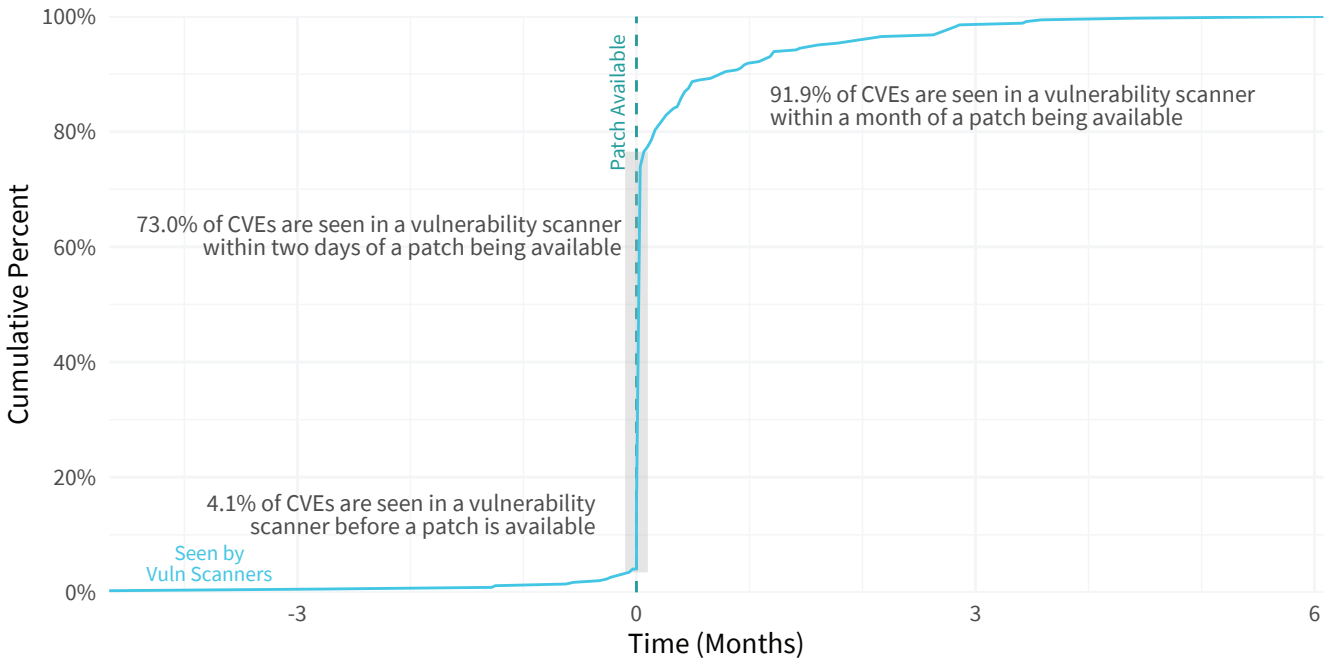
Importance: When a CVE is published, all the pertinent information becomes fully available to both attackers and defenders (though one or both of those groups often have it before that). Prior to a patch, options are limited for mitigating exposure. Once they are available, remediation can begin in earnest.

Observations:

- In great news for defenders, over 80% of vulnerabilities have a patch available prior to, or along with, CVE publication. It's even better news that patch availability generally outpaces exploit code (see Figure 12), but let's not get too far ahead.
- This reflects well on the pervasiveness and effectiveness of coordinated disclosure of vulnerabilities. By the time CVEs become official, defenders usually have what they need to begin remediation.

https://t.me/learningnets

Figure 11: Timing of first detection by vulnerability scanner relative to patch availability



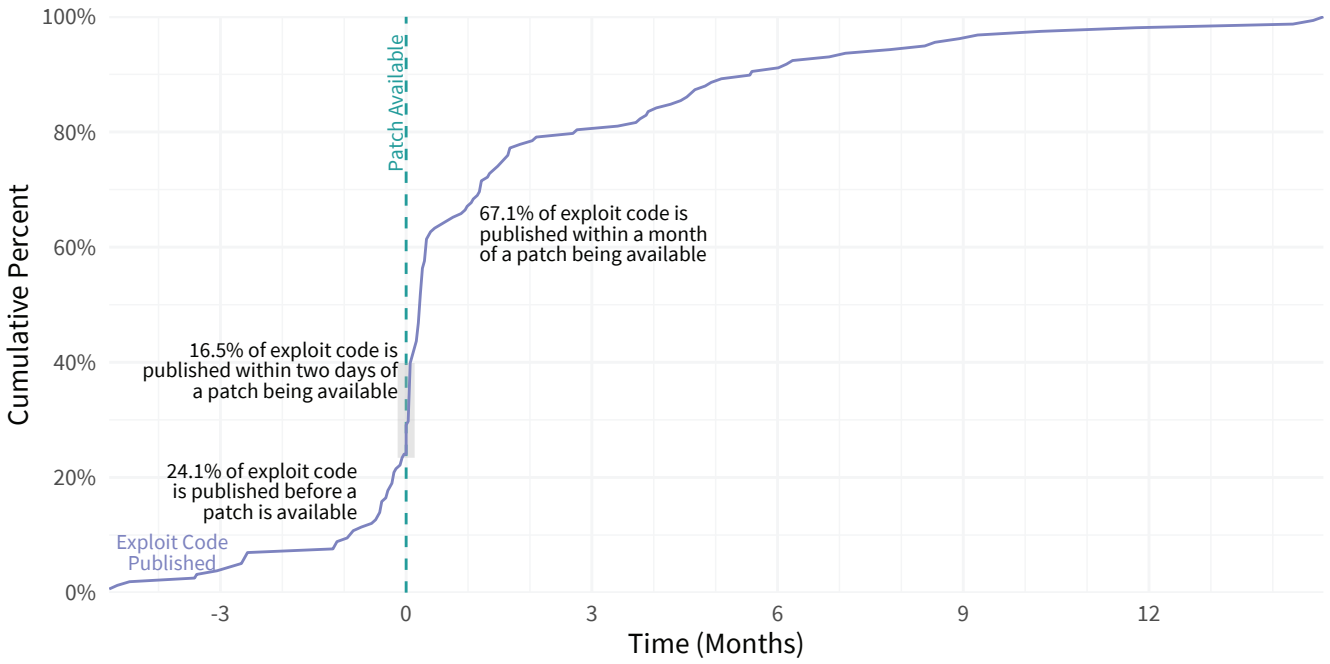
Importance: We said earlier that patch availability marks the point where defenders can begin remediation. The first detection of a vulnerability by a scanner signals that remediation has actually begun. Think of this as a measure of how quickly defenders get off the starting line.

Observations:

- More than 9 out of 10 vulnerabilities have been detected by a vulnerability scanner in a live network environment by the time the patch becomes available. This means defenders (a) know where the vulnerability exists across their assets and (b) have the means (the patch) to begin remediating it.
- Once again, this suggests that coordinated disclosure works as intended. By the time CVEs become official and the patch is available, defenders have begun their work of remediation.
- The time required for the remediation process to complete, however, is another matter. Put that idea in the back of your mind for now, and we'll resume that line of thought later.

https://t.me/learningnets

Figure 12: Timing of exploit code release relative to patch availability



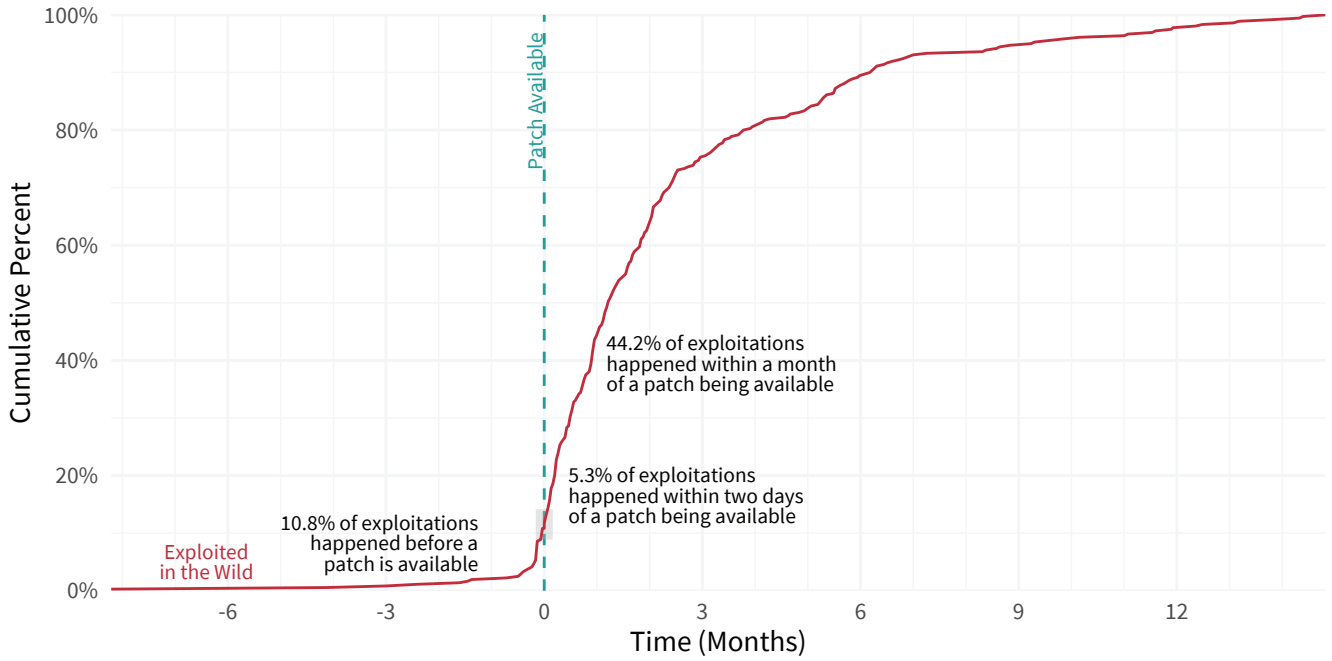
Importance: These milestones represent the basic building blocks for attacker and defender workflows. Attackers add exploit code to their arsenal, and defenders use patches and exploit code to jumpstart remediation.

Observations:

- About a third of vulnerabilities have exploit code published before a patch is made available. This is interesting for two reasons: 1) it contradicts a common argument that exploits enable patch development, and 2) it's the window of time where attackers have a head start on defenders.
- The sharp uptick in exploit code availability right around patch release looks like a causal relationship. But it's more likely indicative of the collective realization that a vulnerability is important and requires action (for good or ill).

https://t.me/learningnets

Figure 13: Timing of first exploitation in the wild relative to patch availability



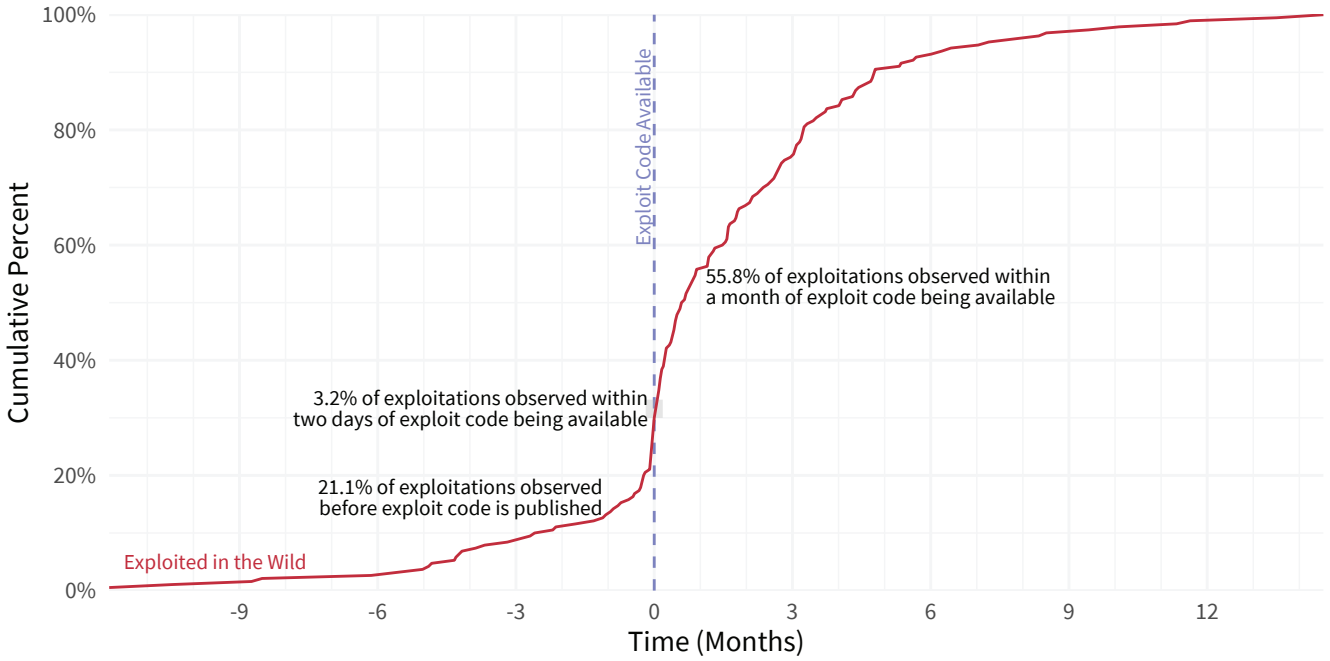
Importance: Exploitation in the wild is the strongest signal we have that attackers are actively probing for and exploiting vulnerabilities in real enterprise environments. Yes, exploit detections can and do stem from red team exercises, but certainly not all of them.

Observations:

- The relationship between patch availability and exploitation implies that defenders typically have some time before they become the target of active exploitation attempts. Using the common enterprise goal of deploying patches within a month of availability, we see that roughly 45% of all exploited vulnerabilities have initial detections in the wild before that window is completed.
- By 6 months after patch availability, 90% of CVEs that were exploited in the wild showed up as detected. The longer a vulnerability is known and patched, the less likely it is to progress from potential to active exploitation in the wild.

https://t.me/learningnets

Figure 14: Timing of first exploitation in the wild relative to exploit code release



Importance: The preceding charts show exploit code and exploitation in the wild separately, which makes them seem like independent events. The data reveals they're highly correlated, and that's a very important fact for defenders racing against the exploitation clock to remember.

Observations:

- Exploit code predates exploitation in the wild for most (70%) exploited CVEs. We knew this from an earlier section, but it's nice to have some numbers around that.
- It's hard not to look at this and jump from correlation to causation. And honestly, that may not be a leap too far. The ramp up of exploitation activity targeting enterprise assets following the release of exploit code makes a compelling case for causality.
- A counterpoint to that, as we've mentioned before, is that many assert that the release of exploit code helps defenders more than attackers because it helps the creation of detection signatures in various defensive measures.
- For now, let's just say: where there's smoke there's fire, and this is smoking.

https://t.me/learningnets

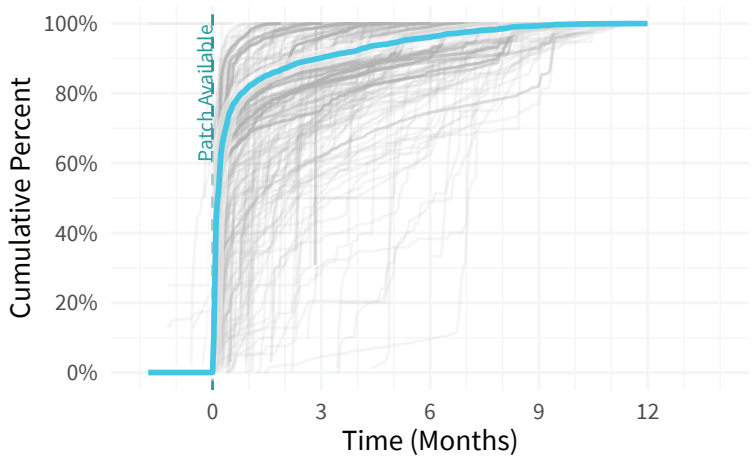
Measuring Momentum in Attack and Defense

Speaking of smoke, this next section is a potential fire starter. We're going to provide some hard data measuring the relationship between attacker and defender momentum in the realm of vulnerability management. To get there, we'll need to establish three overall timelines for vulnerabilities:

1. How quickly vulnerability scanners detect all vulnerable assets
2. How quickly defenders remediate all detected vulnerabilities
3. How quickly exploitation of vulnerabilities spreads in the wild

We'll start by measuring how quickly vulnerabilities are detected by scanners across enterprise environments. Each line in Figure 15 represents detection of a separate vulnerability, progressing from 0% to 100% of affected assets. It's clear that it sometimes takes several months to discover all instances of a vulnerability, while other times, this discovery takes only days. Here, we're not focusing on why these differences exist so much as we're interested in establishing the typical vulnerability detection time. That typical line is represented by the heavy aqua line in the middle of all the others.

Figure 15: Overall timeline of first detection by vulnerability scanners relative to patch availability



Over 80% of vulnerabilities across the environment have been detected within a month.

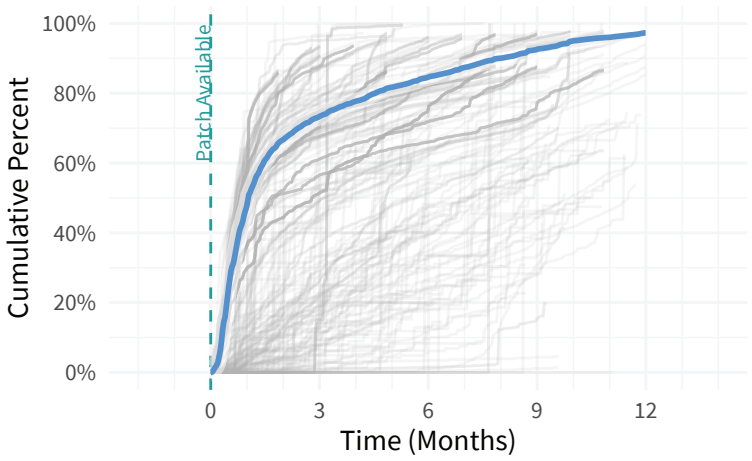
Based on the average scanner detection timeline, it's apparent that the work of finding vulnerable assets begins in earnest after a patch is released. Over 80% of vulnerabilities across the environment have been detected within a month. After that initial Pareto Principle push, scanning all the nooks and crannies of the network to find the remaining 20% usually takes quite a while.

Discovering vulnerabilities is important, but that alone doesn't mitigate risk. To achieve that, VM teams actually have to fix those vulnerabilities. We've analyzed remediation velocity extensively in prior volumes of the P2P series, so long-time readers should be familiar with the concept of survival analysis applied to VM. If you're just joining us, survival curves show the probability that a given vulnerability will be remediated within a particular period of time.

https://t.me/learningnets

In Figure 16, we've used that technique to plot survival curves for each CVE in our dataset. From this, we can see that the aggregate remediation curve crosses the 50% mark about one month after the patch is available. It takes about 5 months to remediate 80% of vulnerabilities across the environment. Again, if you're interested in more stats on that and tons of factors that improve/harm remediation velocity, you have five volumes of awesomeness to explore in prior P2Ps. Also, don't work too hard trying to compare the aggregate line across these figures; we'll make that much easier for you in a moment. Instead, let's just note that remediation of vulnerabilities lags scanner detection: because that's how it works. Find vulns; fix vulns; repeat.

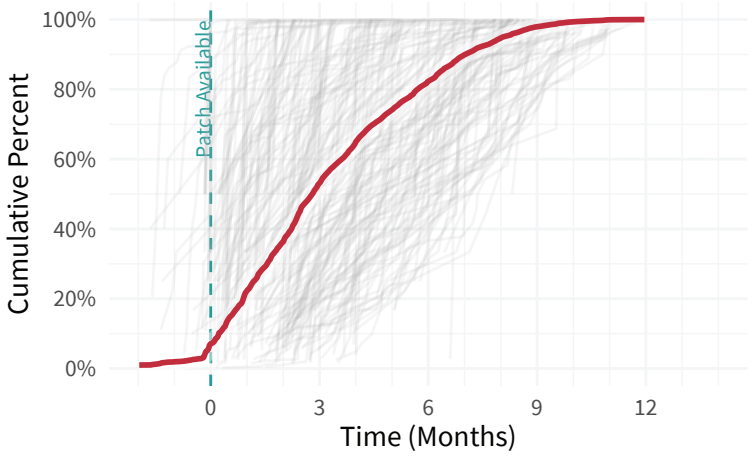
Figure 16: Overall timeline of vulnerability remediation activity relative to patch availability



It takes about 5 months to remediate 80% of vulnerabilities across the environment.

Now, let's jump over to the attacker side of the fence. To create Figure 17, we took the first detection of exploitation in the wild for each vulnerability across the tens of thousands of organizations in our dataset. The lines answer the question: "How long until X% of targeted organizations detects this exploit?" Like the scanner and remediation charts, there's a lot of variability among vulnerabilities, but the heavy red line marks the average.

Figure 17: Overall timeline of spread in exploitation activity in the wild relative to patch availability



It typically takes 6 months for attackers to reach 80% of their target population.

We feel it necessary to add a caveat to Figure 17 so you have the proper context. Recall from Figure 2 that a relatively small proportion of exploits reach widespread prevalence. Some are seen by tens of thousands of organizations; some are seen by less than 10. Whatever the number is, the curves show the progression of exploit activity from the first organization that saw it to the last. Thus, when we say 20% of organizations detect exploits in the first month, that could be 2 firms or 20,000. This doesn't change the facts or interpretation—we just want to make that clear. Speaking of which, here's a nifty factoid: It typically takes 6 months for attackers to reach 80% of their target population.

https://t.me/learningnets

Who Has the Momentum—Attackers or Defenders?

Figure 18 extracts the central scanner, remediation, and exploitation curves from the charts above and puts them in one convenient location. As researchers, we usually study these concepts in isolation, so it's cathartic to be able to view them all together. As much as we love Figure 18, it's mainly a plot device to get to an even more interesting part of the story.

Figure 18: Comparison of vulnerability detection, remediation, and exploitation timelines

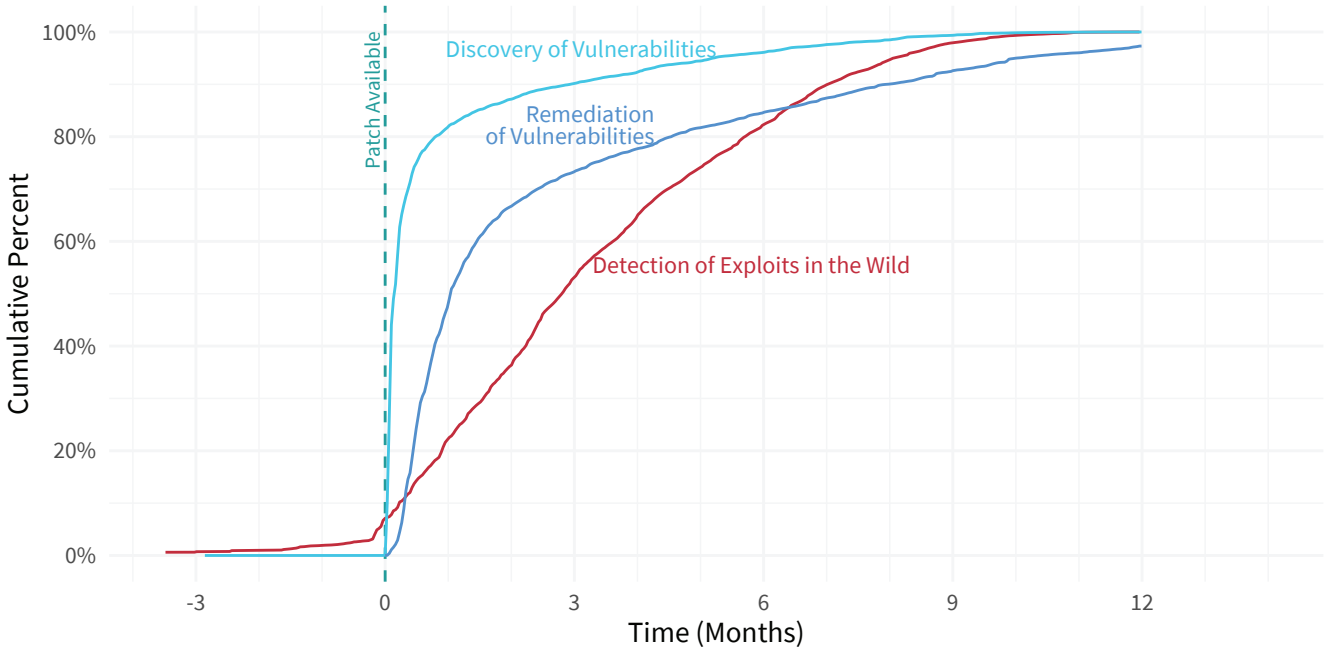
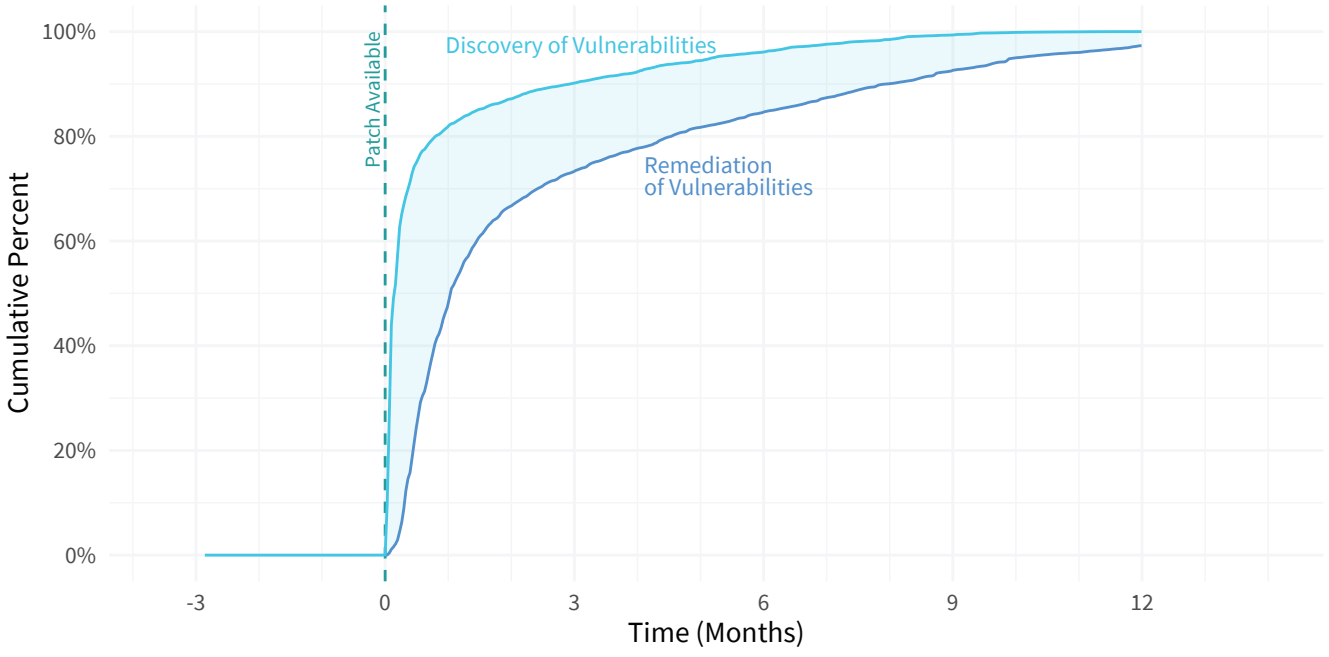


Figure 19 isolates the scanner detection and remediation timelines. The aqua scanner line is the theoretical maximum for vulnerability remediation, and the blue line is the reality of on-the-ground remediation. The space between the idealized and the practical boundaries is an opportunity for defenders to improve remediation velocity by closing the distance between finding and fixing vulnerabilities.

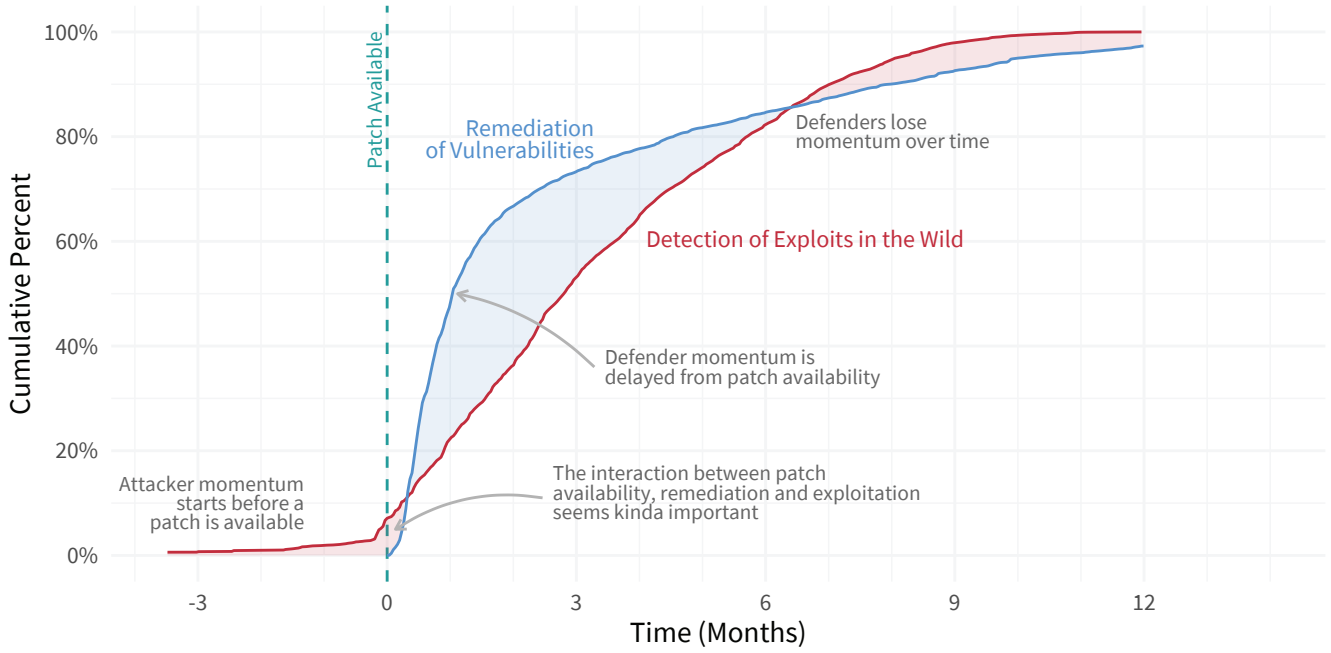
Figure 19: Comparison of vulnerability detection and remediation timelines



https://t.me/learningnets

That brings us to the timeline comparison you've all been waiting for—exploitation vs. remediation. Figure 20 tells us a compelling story of shifting advantage and momentum between attackers and defenders as time progresses. In the time leading up to a vulnerability being patched, attackers have a solid advantage. Armed with the patch, defenders take back the momentum by remediating exposed vulnerabilities across their environment. After that initial push, defender momentum wanes, and attackers regain the long-term advantage. Fascinating, right?

Figure 20: Comparison of vulnerability remediation and exploitation timelines with momentum shifts highlighted



Over the ~15-month period we studied, attackers had the upper hand for about 9 months compared to 6 months for defenders. That being said, we caution against getting caught up in exactly where the timelines intersect in Figure 20 and exactly how much advantage each group attains. As per figures 15–17, there's a ton of variability inherent to these timelines, and what you see in Figure 20 is an amalgamation of them all. Focus instead on the overall picture that's truly worth a thousand words. We're so used to attackers holding the cards that it's incredibly refreshing to see solid evidence that there's a substantial period of time where defenders actually remediate systems faster than attackers exploit them!

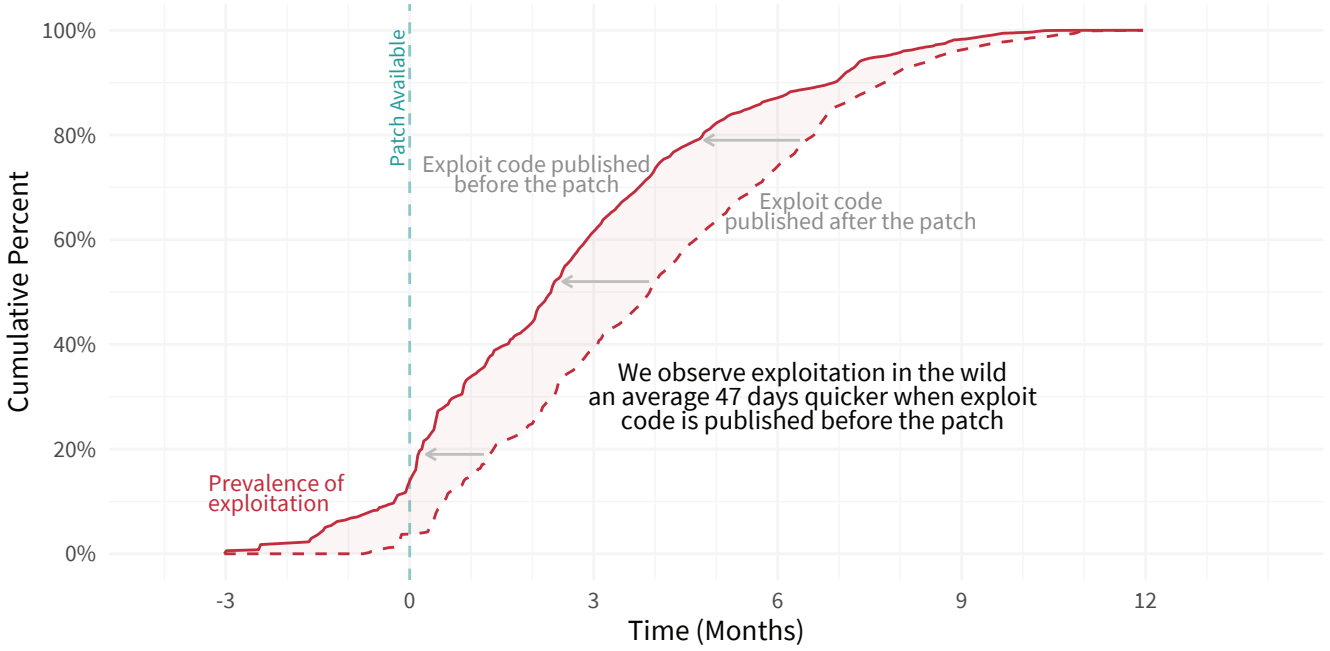
https://t.me/learningnets

Do Early Exploits Shift the Momentum of Attack & Defense?

We know what you're thinking: "How can we shift that momentum even more in the favor of defenders?" And we're right there with you. We're already planning future research devoted to answering that very question. For now, we'd like to ask a much more focused question: Does releasing exploit code before a patch is available measurably alter the relative momentum gains of attackers and defenders?

Figure 21 compares the exploitation timeline for vulnerabilities where the exploit releases after the patch vs. where it releases before the patch. We see exploit activity shifting noticeably to the left when exploit code predates patches in the vulnerability lifecycle. The average magnitude of that shift is 47 days. In other words, dropping exploit code for unpatched vulnerabilities leads to earlier exploitation in the wild.

Figure 21: Comparison of exploitation timelines when exploit code releases after vs. before patch availability

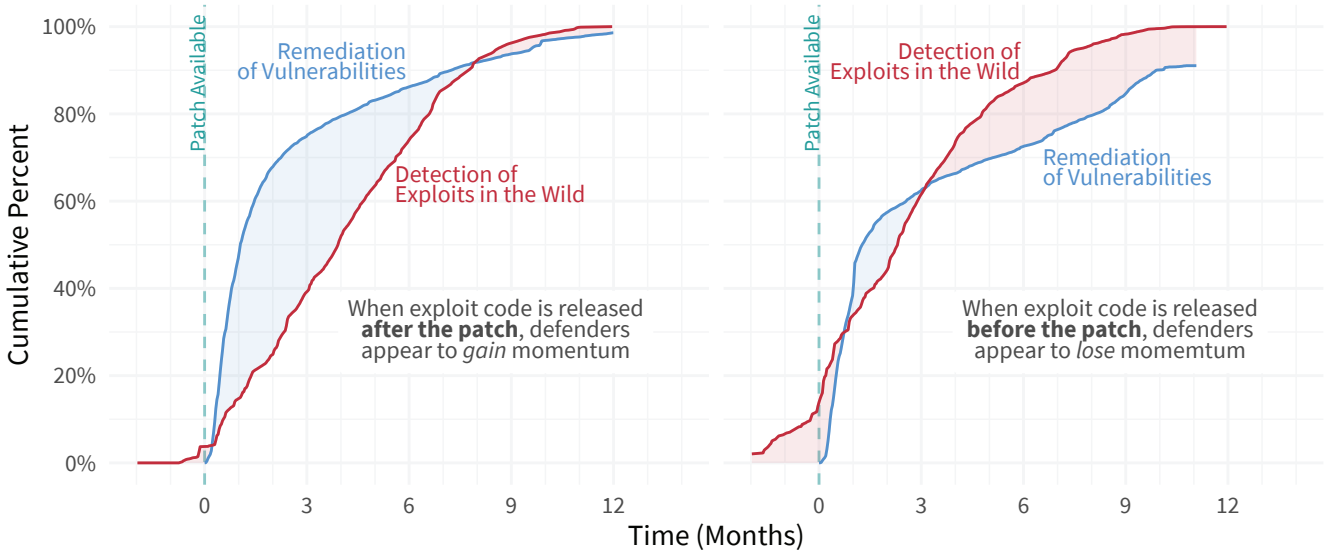


Looking way back at Figure 14, this shouldn't be terribly surprising. Exploitation and exploit code are highly correlated, so a shift in one generally accompanies a shift in the other. What we really want to know is how those pre-patch drops of exploit code shift the relative momentum of exploitation and remediation. Visualizing that is the job of Figure 22.

https://t.me/learningnets

Figure 22 sets up just like Figure 20, plotting the overall timelines for exploitation (red) and remediation (blue). This time, however, we've made one small change and two different charts. In the left chart, we show what those timelines look like for vulnerabilities where the patch predates exploit code. The right chart offers the exploit-before-patch corollary. The difference is unmistakable. The period of defender momentum is drastically reduced when an exploit is released before the patch becomes available. Moreover, attackers control the momentum for a much longer period of time.

Figure 22: Comparison of attacker-defender divide when exploit code releases after vs. before patch availability



This obviously raises some very important—and rather uncomfortable—questions about vulnerability disclosure, remediation, and exploitation. However, before you start jumping to your own conclusions, we ask that you take a few moments to review ours in the next (and last) section.

https://t.me/learningnets

The End… and a Beginning

Good research answers some questions while raising others. So, it's a good sign that we find ourselves in that very situation now. It's hard to believe that after six volumes of heavy analysis of sundry aspects of vulnerability management, there's still no end in sight to the questions we want answered. It seems fitting, therefore, to end this report with an eye towards beginning another. We'll do that in the form of hypotheses pertaining to our most recent discoveries.

We view the shift in defender momentum portrayed in Figure 22 as an incredibly important finding for VM programs, as well as the broader field of cybersecurity. Yet, we also know that it lands smack dab in the middle of one of our industry's oldest and most divisive debates surrounding the disclosure of vulnerabilities and exploits. It's not our intention here to dump fuel on that fire or open old wounds. In fact, we'd really prefer to stick to careful presentation and interpretation of the data. We've presented the data as accurately and transparently as possible. And now, we'd like to offer three possible interpretations of the results shown in Figure 22.

Hypothesis #1: Releasing exploit code early leads to earlier remediation, thus helping defenders. Releasing exploit code can be used to prove to vendors or asset owners that a vulnerability is real or enable admins to remediate prior to a patch. That said, our current data doesn't support this hypothesis. If you compare the two blue remediation timelines in Figure 22, it's evident that remediation does not shift left. On the contrary, it shifts noticeably to the right (longer)—an unexpected fact. We plan to study this over a longer time horizon in future research.

Hypothesis #2: Releasing exploit code early leads to earlier detection, thus helping defenders. It's undeniable that the exploitation timeline shifts left (earlier) in Figure 21 when exploits predate patches. But it's possible that we're just *detecting* exploitation in the wild sooner because exploit code enabled quicker creation and distribution of IPS and AV signatures. To test this hypothesis, we need to incorporate signature creation dates into our timeline and compare those to earliest detections in the wild. If detections spike immediately after deploying signatures, it would suggest that exploitation was occurring all along and earlier detection benefits defenders. If there's a lag between signature release and initial detections, it would suggest the opposite.

Hypothesis #3: Releasing exploit code early facilitates earlier exploitation, thus helping attackers. We suspect this third hypothesis will be dismissed by many because it goes against some strong presuppositions. Regardless of where you stand, let's approach this with cool heads and hard data. The outcomes of testing hypotheses #1 and #2 are big parts of the requirement to test this hypothesis. Beyond that, we'd like to study a much longer time window of remediation and exploitation activity (and all related milestones). We also need to account for possible confounding factors. For instance, maybe the effect we see is limited to only certain types of vulnerabilities or threats.

For our part, we plan to follow the scientific method and collect what evidence we can to support each of these hypotheses (and any others that are posed). But we cannot do that alone and invite the community to conduct their own investigations and/or share relevant data with us, so that we can include it in our continued research on this topic (that will be published later). These topics are important to us all; let's be forces of mutual acceleration rather than opposition.

https://t.me/learningnets

We'd like to tie off this report with some brief concluding thoughts specific to researchers, vendors, and enterprise defenders.

For Researchers: We applaud the efforts of researchers to coordinate the release of vulnerabilities and exploits to coincide with the availability of patches because this helps defenders. Our findings on the lag between patch availability and remediation suggest that it could benefit enterprise defenders even more if additional time was allowed between patch and exploit release in order to reduce exposure.

For Vendors: The fact that key vulnerability milestones often converge around patch release is no coincidence. The data points to coordinated release schedules and, in general, the process seems to be working as intended. We'd encourage vendors to continue supporting coordinated disclosure of vulnerabilities as well as those who engage in it. That seems to work best when researchers are treated as assets rather than adversaries. Furthermore, anything vendors can do to support defenders moving from 'patch available' to deep into the remediation curve as quickly as possible would have large benefits. Based on our prior research, Microsoft appears to be a role model in this regard.

For Defenders:

1. The gap between detection by scanners and remediation of vulnerable assets (Figure 19) represents a lost opportunity for defenders. Anything VM teams can do to shrink the find-fix gap will steal momentum from attackers.

2. The vulnerability lifecycle is incredibly complex. But it's also incredibly connected. The better defenders understand the signs and timelines of exploitation, the better prepared they'll be to increase momentum when needed to stay ahead of attackers.

3. The entwined nature of the exploitation and remediation timelines (Figure 20) is both encouraging and alarming. On the one hand, we've seen that defenders can actually gain momentum over attackers. On the other hand, that advantage is tenuous and can be quickly reversed when external forces like the release of exploit code enter the equation. All of this reinforces the need for strong visibility into exploitation activity as a foundation of risk-based vulnerability management.

PRIORITIZATION TO PREDICTION

VOLUME 6: THE ATTACKER-DEFENDER DIVIDE

KENNA Security

119 Cyentia INSTITUTE

© 2020 Cyentia Institute, LLC. All Rights Reserved. The Cyentia Institute and Kenna Security names and logos are the property of their respective owners.

https://t.me/learningnets