Security Now! #602 - 03-07-17

Let's Spoof!

This week on Security Now!

Countdown to March's patch Tuesday; what was behind Amazon's S3 outage?; why don't I have a cellular connectivity backup?; some additional Cloudflare perspective; Amazon to fight another day over their Voice Assistant's privacy; an examination of the top 9 Android password managers uncovers problems; another fileless malware campaign found in the wild; security improvements in Chrome and Firefox; A proof of concept for BIOS ransomware; a how-to walk-through for return oriented programming; a nifty new site scanning service; Matthew Green compares desktop and mobile security; a bunch of feedback quickies; an incredibly wonderful waste of time accomplishment; the future threat of deliberately fooling AI, and the dark side of automated domain validation certificate issuance.

https://xkcd.com/1807/

Security News

One Week Away from March's Patch Tuesday

- The latest possible 2nd Tuesday
- A single roll-up will fix many pending concerns.

Summary of last week's Amazon S3 Service Disruption

- https://aws.amazon.com/message/41926/
- http://www.theverge.com/2017/3/2/14792442/amazon-s3-outage-cause-typo-internet-server
- http://www.usatoday.com/story/tech/news/2017/03/02/mystery-solved-typo-took-down-big-chunk-web-tuesday/98645754/

AMAZON: We'd like to give you some additional information about the service disruption that occurred in the Northern Virginia (US-EAST-1) Region on the morning of February 28th.

The Amazon Simple Storage Service (S3) team was debugging an issue causing the S3 billing system to progress more slowly than expected. At 9:37AM PST, an authorized S3 team member using an established playbook executed a command which was intended to remove a small number of servers for one of the S3 subsystems that is used by the S3 billing process. Unfortunately, one of the inputs to the command was entered incorrectly and a larger set of servers was removed than intended. The servers that were inadvertently removed supported two other S3 subsystems.

One of these subsystems, the index subsystem, manages the metadata and location information of all S3 objects in the region. This subsystem is necessary to serve all GET, LIST, PUT, and DELETE requests.

The second subsystem, the placement subsystem, manages allocation of new storage and requires the index subsystem to be functioning properly to correctly operate. The placement subsystem is used during PUT requests to allocate storage for new objects.

Removing a significant portion of the capacity caused each of these systems to require a full restart. While these subsystems were being restarted, S3 was unable to service requests. Other AWS services in the US-EAST-1 Region that rely on S3 for storage, including the S3 console, Amazon Elastic Compute Cloud (EC2) new instance launches, Amazon Elastic Block Store (EBS) volumes (when data was needed from a S3 snapshot), and AWS Lambda were also impacted while the S3 APIs were unavailable.

S3 subsystems are designed to support the removal or failure of significant capacity with little or no customer impact. We build our systems with the assumption that things will occasionally fail, and we rely on the ability to remove and replace capacity as one of our core operational processes. While this is an operation that we have relied on to maintain our systems since the launch of S3, we have not completely restarted the index subsystem or the placement subsystem in our larger regions for many years. S3 has experienced massive growth over the last several years and the process of restarting these services and running the necessary safety checks to validate the integrity of the metadata took longer than expected.

The index subsystem was the first of the two affected subsystems that needed to be restarted. By 12:26PM PST, the index subsystem had activated enough capacity to begin servicing S3 GET, LIST, and DELETE requests. By 1:18PM PST, the index subsystem was fully recovered and GET, LIST, and DELETE APIs were functioning normally.

The S3 PUT API also required the placement subsystem. The placement subsystem began recovery when the index subsystem was functional and finished recovery at 1:54PM PST. At this point, S3 was operating normally. Other AWS services that were impacted by this event began recovering. Some of these services had accumulated a backlog of work during the S3 disruption and required additional time to fully recover.

We are making several changes as a result of this operational event. While removal of capacity is a key operational practice, in this instance, the tool used allowed too much capacity to be removed too quickly. We have modified this tool to remove capacity more slowly and added safeguards to prevent capacity from being removed when it will take any subsystem below its minimum required capacity level. This will prevent an incorrect input from triggering a similar event in the future. We are also auditing our other operational tools to ensure we have similar safety checks. We will also make changes to improve the recovery time of key S3 subsystems. We employ multiple techniques to allow our services to recover from any failure quickly. One of the most important involves breaking services into small partitions which we call cells. By factoring services into cells, engineering teams can assess and thoroughly test recovery processes of even the largest service or subsystem.

As S3 has scaled, the team has done considerable work to refactor parts of the service into smaller cells to reduce blast radius and improve recovery. During this event, the recovery time of the index subsystem still took longer than we expected. The S3 team had planned further partitioning of the index subsystem later this year. We are reprioritizing that work to begin immediately.

From the beginning of this event until 11:37AM PST, we were unable to update the individual services' status on the AWS Service Health Dashboard (SHD) because of a dependency the SHD administration console has on Amazon S3. Instead, we used the AWS Twitter feed (@AWSCloud) and SHD banner text to communicate status until we were able to update the individual services' status on the SHD. We understand that the SHD provides important visibility to our customers during operational events and we have changed the SHD administration console to run across multiple AWS regions.

Finally, we want to apologize for the impact this event caused for our customers. While we are proud of our long track record of availability with Amazon S3, we know how critical this service is to our customers, their applications and end users, and their businesses. We will do everything we can to learn from this event and use it to improve our availability even further.

Why don't I (Steve) have a cellular connectivity backup?

- Proportional response.

Amateur (@amateursrg) 3/1/17, 11:52 AM

- @SGgrc Did you read the same response I did?
  Cloudflare totally blamed what they called "ancient software" & acted like this didn't matter.

- I realized that I was relying much more upon the clear responsibility-taking discussion that Nick Sullivan had with Leo on The New Screensavers than on the printed material. What I saw of Cloudflare's written disclosure -- written by John Graham-Cumming -- appeared to be consistent with what nick had said. But I can imagine that someone reading only a more carefully written cporporate post might come away with a different feeling than someone watching a completely candid and off-the-cuff interview.

  If anyone has any doubts, allow me to encourage you to watch Nick at the beginning of TNS: February 25th, Episode #93
  https://twit.tv/shows/new-screen-savers/episodes/93

Amazon's fight over their Voice Assistant privacy ends without any decision
http://www.theverge.com/2017/3/7/14839684/amazon-alexa-first-amendment-case

So… We are not going to obtain any judicial option about search warrants covering past audio obtained by and queries served by Amazon's Voice Assistant technology.

In documents filed last Monday, defendant James Andrew Bates said that he was willing to allow law enforcement officials to review information contained on his Amazon Echo speaker, before the company handed the data over on Friday. Bates has pleaded not guilty to the murder of Victor Collins, who was found dead in Bates' hot tub in November 2015.

Police had issued a warrant to seize subscriber and account information from Bates' Echo, as well as all communication and transaction history from the device. Amazon provided the former, but argued against providing communication data, claiming that voice interactions with Alexa were protected by the First Amendment. That includes Alexa's replies to a user — Amazon claims that ranked search results are "constitutionally protected opinion." Precedent for that argument was set by a 2014 case in which Google search results were classified as "free speech" by a San Francisco court, after a news website complained that its own pages were too far down the company's listings.

Amazon argued that police didn't have enough of a compelling argument in Bates' case for it to hand over the data, with officials unable to prove that any potential information would not be available anywhere else. It remains to be seen whether Bates' Echo does indeed have any pertinent information — a hearing is scheduled for [Tomorrow], Wednesday this week. The defendant's acquiescence also means that we don't yet have a definitive answer on whether this sort of [IoT acquired data] Alexa is indeed protected by the First Amendment.

Security of many Password Manager Apps found wanting

- https://team-sik.org/trent_portfolio/password-manager-apps/

We are TeamSIK (Security Is Key) – a group of people interested in IT-Security from Fraunhofer Institute for Secure Information Technology (Darmstadt, Germany). We meet up regulary in our spare time. Our main motivation is to work on interesting security-related projects for fun with the goal of exposing security issues. Currently, our main areas of interest are Android applications.

Closely examined the top 9 password managers on Android

- MyPasswords
- Informaticore Password Manager
- LastPass Password Manager
- Keeper Password Manager
- F-Secure KEY Password Manager
- Dashlane Password Manager
- Hide Pictures Keep Safe Vault
- Avast Passwords
- 1Password

ABOUT THE PROJECT:
There are different policies for the generation of secure passwords. However, one of the biggest challenges is to memorize all these complex passwords. Password manager applications are a promising way of storing all sensitive passwords cryptographically secure. Accessing these passwords is only possible if the user enters a secret master password. At first sight, the requirements for a password manager application seem simple: Storing the passwords of a user centralized in a secure and confidential way. However, how is the reality on mobile, password manger applications, especially on Android? Applications vendors advertise their password manager applications as "bank-level" or "military-grade" secure. However, can users be sure that their secrets are actually stored securely? Despite the vendors' claims, is it nevertheless possible to obtain access to the stored credentials?

In order to answer these questions, we performed a security analysis on the most popular Android password manager applications from the Google Play Store based on download count. The overall results were extremely worrying and revealed that password manager applications, despite their claims, do not provide enough protection mechanisms for the stored passwords and credentials. Instead, they abuse the users` confidence and expose them to high risks.

We found several implementation flaws resulting in serious security vulnerabilities. Some applications stored the entered master password in plaintext or implemented hard-coded crypto keys in the program code. Consequently, attackers can easily circumvent the crypto algorithm altogether and thereby gain access to all of the user's data. In other cases, we could simply access all "securely protected passwords/credentials" with the help of an additional app. Once installed on the device, this malicious app extracts all passwords/credentials in plaintext and sends them to the attacker. In yet another case, we could use a so-called data residue attack to access the master key of an application. In most of the cases, no root permissions were required for a successful attack that gave us access to sensitive information such as the aforementioned master password. Furthermore, many of the apps completely ignore the problem of clipboard

sniffing, meaning that there is no cleanup of the clipboard after credentials have been copied into it.

While this shows that even the most basic functions of a password manager are often vulnerable, these apps also provide additional features, which can, again, affect security. We found that, for example, auto-fill functions for applications could be abused to steal the stored secrets from the password manager application using “hidden phishing” attacks. For a better support of auto-filling password forms in web pages, some of the applications provide their own web browsers. These browsers are an additional source of vulnerabilities, such as privacy leakage.

All of our 26 findings are provided in detail in the following...

LastPass:
The Android app of LastPass comes with an access control mechanism, which prohibits arbitrary usage of its functionality. By default, the user is asked to enter his master password in order to gain access to the application. Due to its enforced complex requirements a user can easily get frustrated entering the master password over and over again. Therefore, LastPass offers to substitute the master password with a PIN mechanism. Thereby the user can agree to save the master password on the device and shift all access control to a PIN, which can be 4-14 digits long. The master key and the PIN are symmetrically encrypted and stored in a shared preferences file in the local app folder. The key/PIN are stored encrypted. The key for encrypting/decrypting the credentials is hard coded into the application’s source code.

For stealing the encrypted master key/PIN, we assume the attacker gains access to the device (evil maid attack). The first approach reading out the LPAndroid.xml content is working without a root exploit, but not on each Android version. The second approach requires a root exploit (there exists different types for different Android versions).

The Emergence of a disturbingly clever file-free malware

- http://thehackernews.com/2017/03/powershell-dns-malware.html
- http://www.darkreading.com/attacks-breaches/attackers-employ-sneaky-new-method-to-control-trojans/d/d-id/1328320

Cisco's Talos security research group discovered a new and sophisticated attack which launches from an eMail phishing malicious Word document and successfully executes a Windows PowerShell backdoor by communicating with its command-and-control servers through a series of DNS requests.

TITLE: "Covert Channels and Poor Decisions: The Tale of DNSMessenger"
http://blog.talosintelligence.com/2017/03/dnsmessenger.html

The DNSMessenger attack is completely Fileless -- it does not involve writing files to the targeted system -- it uses DNS TXT messaging capabilities to fetch malicious PowerShell commands stored remotely as DNS TXT records... thus rendering it invisible to existing anti-malware defenses.

As we know, PowerShell is an extremely capable scripting language built into Windows to allow for the automation of most system administration tasks.

The Word document launches a VBA -- Visual Basic for Applications -- macro which executes a self-contained PowerShell script to initiate the RAT -- Remote Access Trojan -- on the target system.

And everything is done in memory, without writing any malicious files to the system's disk.

The VBA script unpacks a compressed second stage of PowerShell, which checks for several parameters of the target environment including the privileges of the logged-in user and the version of PowerShell installed on the target system. This information is used to ensure persistence on the infected host by changing the Windows Registry and installing a third stage PowerShell script that contains a simple backdoor.

The backdoor is added to the Windows Management Instrumentation (WMI) database, if the victim has administrative access, allowing the malware backdoor to stay persistent on the system even after a reboot.

The backdoor is a script that establishes a 2-way communications channel over the Domain Name System (DNS) using DNS TXT records.

The backdoor periodically sends DNS queries to one of a series of domains hard-coded in its source code. As part of those requests, it retrieves the domain's DNS TXT record, which contains further PowerShell commands that are executed but never written to the local system.

Chrome further tightens up security on macOS

- https://security.googleblog.com/2017/03/expanding-protection-for-chrome-users.html

Google's Blog Posting:
Safe Browsing is broadening its protection of macOS devices, enabling safer browsing experiences by improving defenses against unwanted software and malware targeting macOS. As a result, macOS users may start seeing more warnings when they navigate to dangerous sites or download dangerous files (example warning below).

As part of this next step towards reducing macOS-specific malware and unwanted software, Safe Browsing is focusing on two common abuses of browsing experiences: unwanted ad injection, and manipulation of Chrome user settings, specifically the start page, home page, and default search engine. Users deserve full control of their browsing experience and Unwanted Software Policy violations hurt that experience.

The recently released Chrome Settings API for Mac gives developers the tools to make sure users stay in control of their Chrome settings. From here on, the Settings Overrides API will be the only approved path for making changes to Chrome settings on Mac OSX, like it currently is on Windows. Also, developers should know that only extensions hosted in the Chrome Web Store are allowed to make changes to Chrome settings.

Starting March 31 2017, Chrome and Safe Browsing will warn users about software that attempts to modify Chrome settings without using the API.

For many more nitty-gritty details:
https://developer.chrome.com/extensions/settings_override

Firefox: Containers Come to Test Pilot

- https://hacks.mozilla.org/2017/03/containers-come-to-test-pilot/

What are containers?
https://blog.mozilla.org/tanvi/2016/06/16/contextual-identities-on-the-web/
Tanvi Vyas / Security Engineer
Title: "Contextual Identities on the Web"

The Containers Feature in Firefox Nightly enables users to login to multiple accounts on the same site simultaneously and gives users the ability to segregate site data for improved privacy and security.

We all portray different characteristics of ourselves in different situations. The way I speak with my son is much different than the way I communicate with my coworkers. The things I tell my friends are different than what I tell my parents. I'm much more guarded when withdrawing money from the bank than I am when shopping at the grocery store. I have the ability to use multiple identities in multiple contexts. But when I use the web, I can't do that very well. There is no easy way to segregate my identities such that my browsing behavior while shopping for toddler clothes doesn't cross over to my browsing behavior while working. The Containers feature I'm about to describe attempts to solve this problem: empowering Firefox to help segregate my online identities in the same way I can segregate my real life identities.

With Containers, users can open tabs in multiple different contexts – Personal, Work, Banking, and Shopping. Each context has a fully segregated cookie jar, meaning that the cookies, indexeddb, localStorage, and cache that sites have access to in the Work Container are completely different than they are in the Personal Container. That means that the user can login to their work twitter account on twitter.com in their Work Container and also login to their personal twitter on twitter.com in their Personal Container. The user can use both accounts in side-by-side tabs simultaneously. The user won't need to use multiple browsers, an account switcher[1], or constantly log in and out to switch between accounts on the same domain.

Note that the inability to efficiently use "Contextual Identities" on the web has been discussed for many years. The hard part about this problem is figuring out the right User Experience and answering questions like:

- How will users know what context they are operating in?
- What if the user makes a mistake and uses the wrong context; can the user recover?
- Can the browser assist by automatically assigning websites to Containers so that users don't have to manage their identities by themselves?
- What heuristics would the browser use for such assignments?

We don’t have the answers to all of these questions yet, but hope to start uncovering some of them with user research and feedback. The Containers implementation in Nightly Firefox is a basic implementation that allows the user to manage identities with a minimal user interface.

Note that we've had extensive similar discussions with the SQRL project. I ended up implementing "Alternative IDs" so that users would not need to create entirely separate SQRL identities.

What is the security model provided by Containers?

The security enhancements of Containers in Nightly and Test Pilot is common across both versions, and are based on a modification to the browser’s Same Origin Policy (SOP).

As we know, the Same Origin Policy ensures that documents and data from distinct origins are isolated from each other. It is a critical browser security mechanism that prevents content from one site from being read or altered by another, potentially malicious site.

A page's Origin is defined by the tuple composed of the Scheme, Host & Port. So, for example: (https, www.grc.com, 443). Containers add an extra parameter – a userContextId integer – to the normal (scheme, host, port) tuple that defines an origin. So, an origin is now defined as (userContextId, scheme, host, port).

When a user visits Gmail in a Work container tab, the browser performs the SOP check against (2, https, mail.google.com, 443). When the same user visits Gmail in a Personal container tab, the browser performs the SOP check against (1, https, mail.google.com, 443).

Containers separate cookies, localStorage, indexedDB, and cache data from each other and from the Default container in Firefox. So, when a user visits their email site in a Work container tab, the browser sets its cookies only in the Work container. If they then visit their email site in a Personal container, the origin that has their cookies doesn’t match and the user is therefore “signed out”.

Because cookies are not shared across containers, cookie-based attacks in one container are unsuccessful against cookies stored in another container. Similarly, cookie-based tracking only tracks a single container – it does not track the user’s entire browsing.

Many privacy and security mechanisms can be realized by including more keys in the origin check. Because of this, Gecko has added attributes to the origin called OriginAttributes. In addition to Containers, this allows us to implement features like Private Browsing Mode, First Party Isolation, and potentially the proposed Suborigins standard.

UEFI Ransomware: Full Disclosure at Black Hat Asia
https://www.cylance.com/en_us/blog/uefi-ransomware-full-disclosure-at-black-hat-asia.html

During last month's RSA 2017 conference, the guys from Cylance gave a live demo of Cylance's UEFI Ransomware proof of concept. In that demo they successfully exploited a system with an Intel Skylake CPU running Microsoft Windows 10 Enterprise (1607) with all updates. They enabled all available security features including Secure Boot, Virtual Secure Mode (VSM), and Device Guard (with its default policy).

Despite all of those protections and mitigations based on virtualization technologies in Windows 10... systems remain vulnerable to UEFI-based attacks from System Management Mode (SMM). Because SMM allows direct access to physical memory, it's possible to bypass the virtualization layer of isolation (Intel VT-x). This allows an unprivileged application to install persistent malware, such as ransomware, invisibly into a motherboard's firmware.

At the upcoming Black Hat Asia - March 28-31, 2017 in Singapore - they will be disclosing all of the details of the vulnerabilities they exploited in their talk titled: 'UEFI Firmware Rootkits: Myths and Reality'.

Gargoyle: A memory Scanning Evasion Technique

- https://jlospinoso.github.io/security/assembly/c/cpp/developing/software/2017/03/04/gargoyle-memory-analysis-evasion.html

A very nicely written-up and detailed example of a practical working ROP (Return Oriented Programming) exploit on Windows demonstrating how an attacker can arrange to execute pure non-executable data.

New service: URLscan.io (Beta)

- CNN: https://urlscan.io/result/14880285-3fcf-4be2-9992-978210b7961f#summary
- GRC: https://urlscan.io/result/c0300b77-b96f-4e98-8752-15140a0e5d22#summary

Requests, Ads Blocked, Malicious, Secure, IPv6, Domains, Subdomains, IPs, Countries, Sizes, Cookies, and much much more.

Matthew Green: Secure computing for journalists
https://blog.cryptographyengineering.com/2017/03/05/secure-computing-for-journalists/

This morning on Twitter, Buzzfeed editor Miriam Elder asks the following question:

- Possibly stupid question: is the Signal desktop client as secure as the mobile app?
  — Miriam Elder (@MiriamElder) March 3, 2017

No, this is not a stupid question. Actually it's an extremely important question, and judging by some of the responses to this Tweet there are a lot of other people who are confused about the answer.

Since I couldn't find a perfect layperson's reference anywhere else, I'm going to devote this post to providing the world's simplest explanation of why, in the threat model of your typical journalist, your desktop machine isn't very safe. And specifically, why you're safer using a modern mobile device — and particularly, an iOS device — than just about any other platform.

A brief caveat: I'm a cryptographer, not a software security researcher. However, I've spent the past several years interacting with [many] folks [who live in this space]. I'm pretty confident that they agree with this advice.

What's wrong with my laptop/desktop machine?

Sadly, most of the problem is you.

If you're like most journalists — and really, most professionals — you spend less than 100% of your time thinking about security. You need to get work done. When you're procrastinating from work, you visit funny sites your friends link you to on Facebook. Then you check your email. If you're a normal and productive user, you probably do a combination of all these things every few minutes, all of which culminates in your downloading some email attachment and (shudder) opening it in Word.

- You can't tell journalists "just don't open attachments." They will ignore you. Journos open attachments from strangers for a living.
  — Eva (@evacide) September 12, 2016

Now I'm not trying to shame you for this. It's perfectly normal, and indeed it's necessary if you want to get things done. But in the parlance of security professionals, it also means you have a huge attack surface.

In English, this means that from the perspective of an attacker there are many different avenues to compromise your machine. Many of these aren't even that sophisticated. Often it's just a matter of catching you during an unguarded moment and convincing you to download an executable file or an infected Office document. A compromised machine means that every piece of software on that machine is also vulnerable.

If you don't believe this works, head over to Google and search for "Remote Access Trojans". There's an entire commercial market for these products, each of which allows you to remotely control someone else's computer. These off-the-shelf products aren't very sophisticated: indeed, most require you to trick your victim into downloading and running some executable attachment. Sadly, this works on most people just fine. And this is just the retail stuff. Imagine what a modestly sophisticated attacker can do.

I do some of those things on my phone as well. Why is a phone better?

Classical (desktop and laptop) operating systems were designed primarily to support application developers. This means they offer a lot of power to your applications. An application like Microsoft Word can typically read and write all the files available to your account. If Word becomes compromised, this is usually enough to pwn you in practice. And in many cases, these

applications have components with root (or Administrator) access, which makes them even more dangerous.

Modern phone operating systems like Android and iOS were built on a different principle. Rather than trusting apps with much power, each app runs in a "sandbox" that (mainly) limits it to accessing its own files. If the sandbox works, even a malicious application shouldn't be able to reach out to touch other apps' files or permanently modify your system. This approach — combined with other protections such as in-memory code signing, hardware secret storage and routine use of anti-exploitation measures — makes your system vastly harder to compromise.

Of course, sandboxing isn't perfect. A compromised or malicious app can always access its own files. More sophisticated exploits can "break out" of the sandbox, typically by exploiting a vulnerability in the operating system. Such vulnerabilities are routinely discovered and occasionally exploited.

The defense to this is twofold: (1) first, run a modern, up-to-date OS that receives security patches quickly. And (2) avoid downloading malicious apps. Which brings me to the main point of this post.

Why use iOS?

The fact of the matter is that when it comes to addressing these remaining issues, Apple phone operating systems (on iPhones and iPads) simply have a better track record.

Since Apple is the only manufacturer of iOS devices, there is no "middleman" when it comes to monitoring for iOS issues and deploying iOS security updates. This means that the buck stops at Apple — rather than with some third-party equipment manufacturer. Indeed, Apple routinely patches its operating systems and pushes the patches to all supported users — sometimes within hours of learning of a vulnerability (something that is relatively rare at this point in any case).

Of course, to be fair: Google has also become fairly decent at supporting its own Android devices. However, to get assurance from this process you need to be running a relatively brand new device and it needs to be manufactured by Google. Otherwise you're liable to be several days or weeks behind the time when a security issue is discovered and patched — if you ever get it. And Google still does not support all of the features Apple does, including in-memory code signing and strong file encryption.

Apple also seems to do a relatively decent job at curating its App Store, at least as compared to Google. And because those apps support a more modern base of phones, they tend to have access to better security features, whereas Android apps more routinely get caught doing dumb stuff for backwards compatibility reasons.

Finally, every recent Apple device (starting with the iPhone 5S and up) also includes a specialized chip known as a "Secure Enclave Processor". This hardened processor assists in securing the boot chain — ensuring that nobody can tamper with your operating system. It can also protect sensitive values like your passwords, ensuring that only a password or fingerprint can access them.

A few Android phones also offer similar features as well. However, it's unclear how well these are implemented in contrast to Apple's SEP. It's not a bet I would choose to take.

So does using iOS mean I'm perfectly safe?

Of course not. Unfortunately, computer security today is about resisting attacks. We still don't quite know how to prevent them altogether.

Indeed, well-funded attackers like governments are still capable of compromising your iOS device (and your Android, and your PC or Mac). Literally the only question is how much they'll have to spend doing it.

Here's one data point. Last year a human rights activist in the UAE was targeted via a powerful zero day exploit, likely by his government. However, he was careful. Instead of clicking the link he was sent, the activist sent it to the engineers at Citizenlab who reverse-engineered the exploit. The resulting 35-page technical report by Lookout Security and Citizenlab is a thing of terrifying beauty: it describes a chain of no less than three previously unpublished software exploits, which together would have led to the complete compromise of the victim's iPhone.

But such compromises don't come cheap. It's easy to see this kind of attack costing a million dollars or more. This is probably orders of magnitude more than it would cost to compromise the typical desktop user. That's important. Not perfect, but important.

You're telling me I have to give up my desktop machine?

Not at all. Or rather, while I'd love to tell you that, I understand this may not be realistic for most users.

All I am telling you to do is to be thoughtful. If you're working on something sensitive, consider moving the majority of that work (and communications) to a secure device until you're ready to share it. This may be a bit of a hassle, but it doesn't have to be your whole life. And since most of us already carry some sort of phone or tablet in addition to our regular work computer, hopefully this won't require too much of a change in your life.

You can still use your normal computer just fine, as long as you're aware of the relative risks. That's all I'm trying to accomplish with this post.

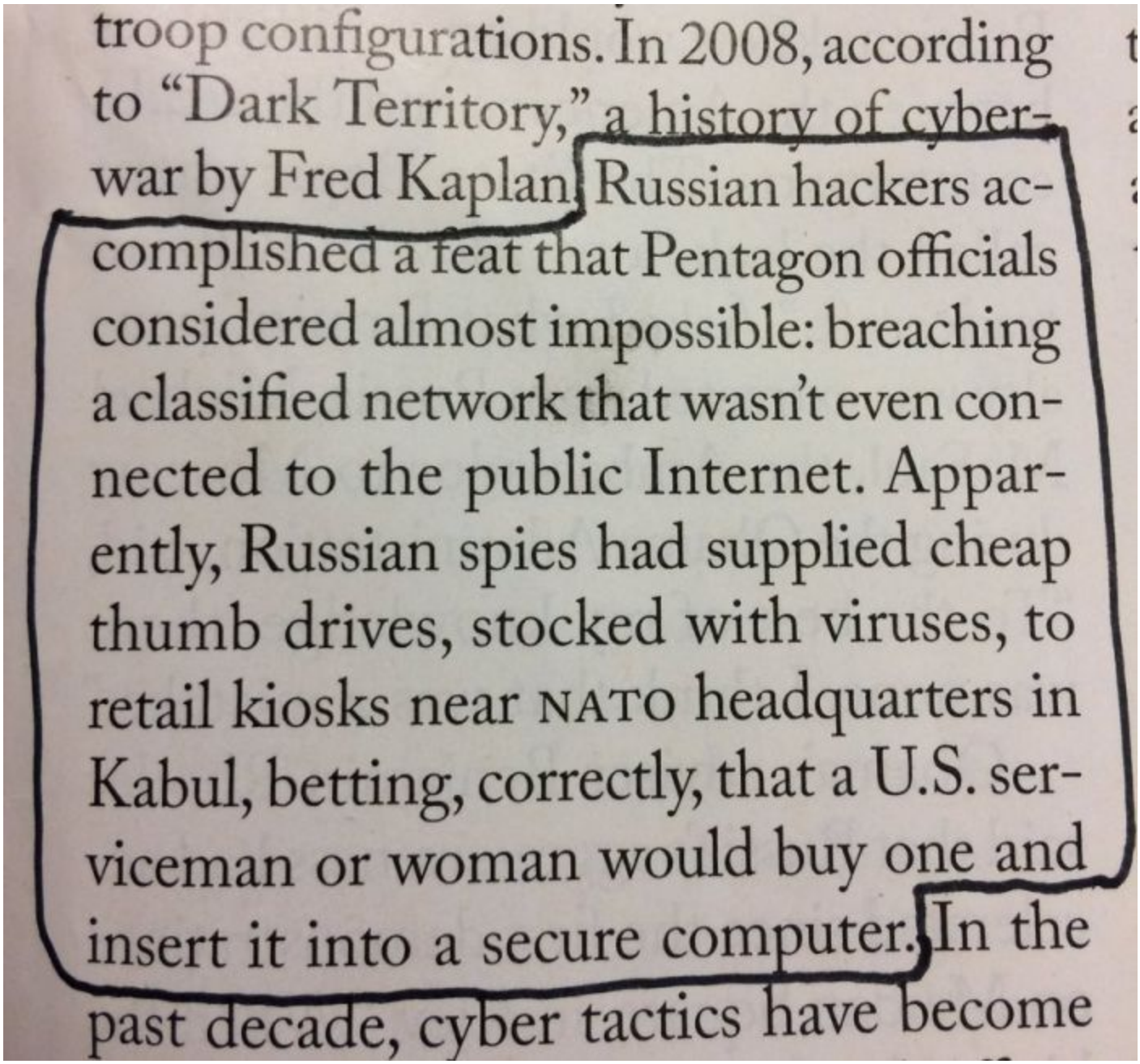
In conclusion

I expect that many technical people will find this post objectionable, largely because they assume that with their expertise and care they can make a desktop operating system work perfectly safely. And maybe they can! But that's not who this post is addressed to.

And of course, this post still only scratches the surface of the problem. There's still the problem of selecting the right applications for secure messaging (e.g., Signal and WhatsApp) and finding a good secure application for notetaking and document collaboration and so on.

But hopefully this post at least starts the discussion.

Mathew Clayton (@Mathew__Clayton) 3/3/17, 2:09 AM
Loved this detail from the New Yorker's piece on Russian cybercrime
pic.twitter.com/mpfYySL2zH


troop configurations. In 2008, according to "Dark Territory," a history of cyber-war by Fred Kaplan, Russian hackers accomplished a feat that Pentagon officials considered almost impossible: breaching a classified network that wasn't even connected to the public Internet. Apparently, Russian spies had supplied cheap thumb drives, stocked with viruses, to retail kiosks near NATO headquarters in Kabul, betting, correctly, that a U.S. serviceman or woman would buy one and insert it into a secure computer. In the past decade, cyber tactics have become

Ben Riggs (@bennyriffins) 3/6/17, 2:27 PM

- @SGgrc Steve - What software do you use to monitor your cox cable connection at home?

Dan Kutka (@dkutka25) 3/3/17, 8:56 PM

- @SGgrc would love to hear your toolkit for keeping an eye on your home Internet traffic on the podcast sometime

- Ping Plotter / Ping Plotter Pro : https://www.pingplotter.com/

Richard Covington (@RKCovington) 3/5/17, 7:21 PM

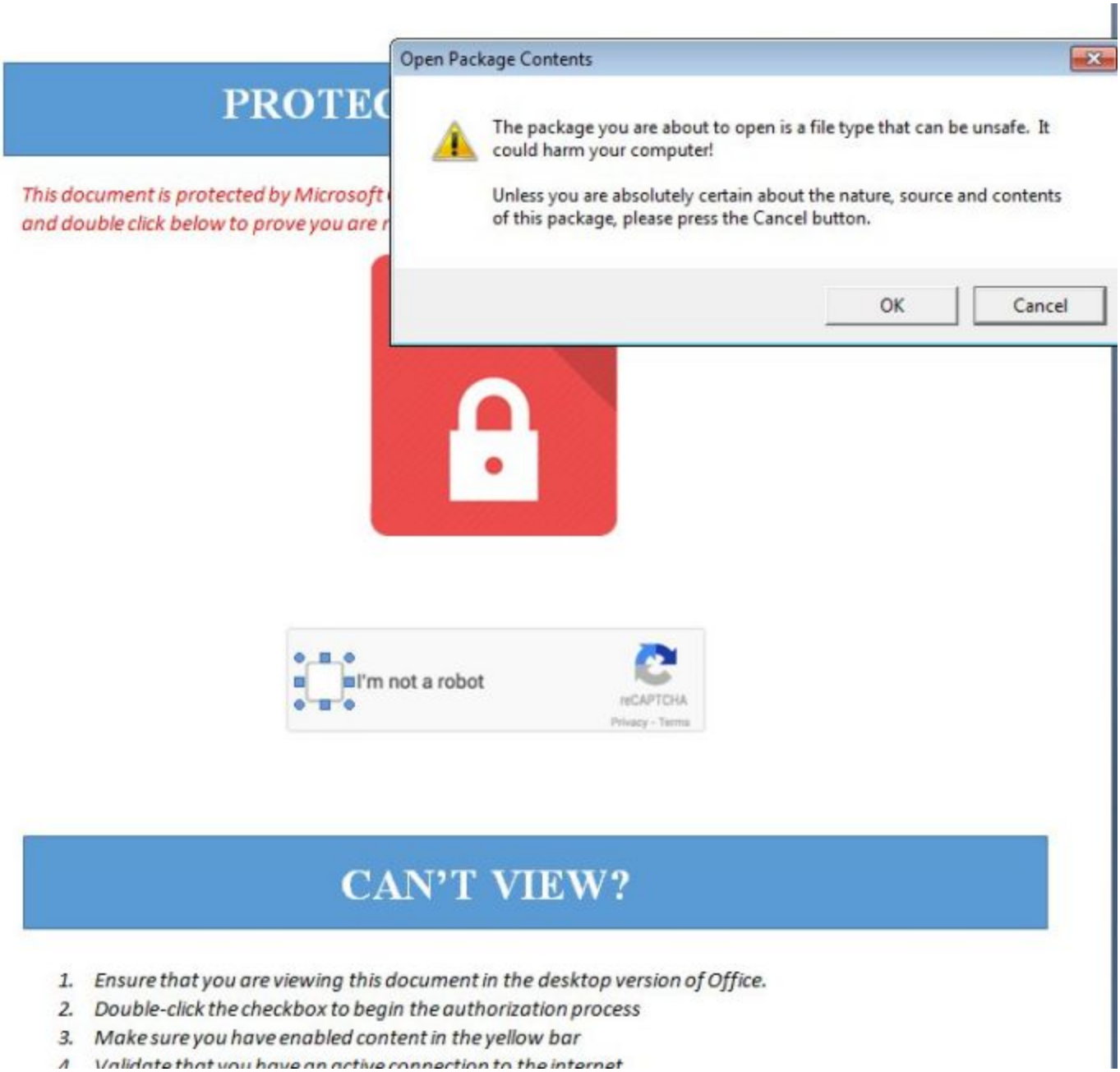
- @SGgrc SN#600 - one can disable the microphone on computers - just plug in the male mic plug (with the wires tied together) in the mic jack

Kevin Bryant (@HackVector) 3/6/17, 5:58 PM

- GRC's "Shields Up" will only scan the first 1056 ports.
  Are there no one-click Scan-All-My-Ports website? CC:@SGgrc

Simon Zerafa (@SimonZerafa) 3/2/17, 12:41 PM

- @SGgrc Beware of those "I am not a robot" tests in unexpected places
  JaromirHorejsi (@JaromirHorejsi) 3/2/17, 6:54 AM
  Double click on "I'm not a robot" to activate powershell payload
  https://virustotal.com/en/file/d5d75119a30517b702fdf89a5534f0dfa6ed46f62fcd1a4b7b87cd1f12948a6e/analysis/



The coolest waste of time... ever:

http://codegolf.stackexchange.com/questions/88783/build-a-digital-clock-in-conways-game-of-life/

I was a high school freshman when, in the October 1970 issue of Scientific American, Martin Gardener's "Mathematical Games" column…

https://en.wikipedia.org/wiki/Conway's_Game_of_Life

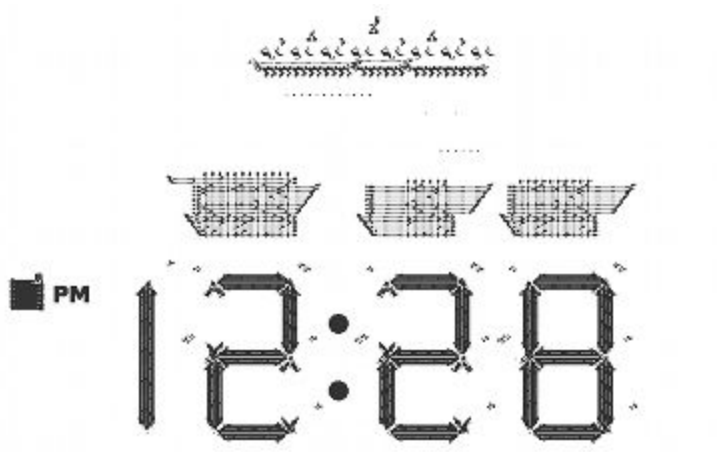
John Horton Conway

- Any live cell with fewer than two live neighbours dies, as if caused by underpopulation.
- Any live cell with two or three live neighbours lives on to the next generation.
- Any live cell with more than three live neighbours dies, as if by overpopulation.
- Any dead cell with exactly three live neighbours becomes a live cell, as if by reproduction.

What??

- Gliders, the speed of light ('C'), Glider Guns, puffer-type breeders that move along leaving glider guns in their wake... which then, of course, begin emitting gliders.
- Gliders move diagonally.
- Lightweight spaceships move horizontally.
- A 2x2 is a block. It's stable. So are Beehives, loafs, boats and tubs.
- A 1x3 strip is a "blinker".

A digital clock built in Conway's Game Of Life…



SpinRite

Jordan (@Rand0mByteZ) March 1, 7:30am

Hey Steve, [SpinRite Testimonial] - The CEO of my agency, recently brought in her personal laptop, which held her recently created but NOT filed tax return.

The laptop was basically on life support, I don't even think Dr. House could have brought this thing back.

The OS loaded but you couldn't do anything. Right clicking took over a minute to register. The laptops hard drive was just replaced not even a year prior. But, I decided to break out SpinRite. I started a scan at level 2. The scan got to 0.14% complete and reported corrupt bits. At 0.16% I realizing that this called for the big guns. So I stopped the scan, and turned it up to 11 (restarted on level 4). I kicked off the level 4 scan and let it run again. It got to 0.19% but it was running very slowly. So I stopped the scan and thought to try it on level 1. Bingo....Level 1 found and corrected the corrupt bits and the rest of the drive was scanned, checked, and marked good.

Rebooting the machine, the OS came up normal and the laptop was running as if it were a new drive all over again. The CEO got to file her taxes, which made me look like a super hero. Upon payment for my services, there will be another yaba-daba-doo coming your way. Thanks again for the best HDD recovery tool out there.....I'm not sure why your competition even tried.... :) Thanks, Jordan

The Threat presented by deliberately fooling AI.

AI is all the rage now. We have crazy computing power, insane storage, and massive connectivity. What is guaranteed to happen is that we will quickly grow to depend upon these technologies and to incorporate them into our lives, as we have email, social networking, and most other technology that is useful on a daily basis. And insufficient attention will be given to the question of how it might fail. The problem is, everything we have learned about security suggests that things generally work well only so long as no one is trying to DELIBERATELY subvert our technologies... because our technologies are still not being designed with an eye to malicious abuse.

Attacking machine learning with adversarial examples
https://openai.com/blog/adversarial-example-research/
"Adversarial examples" are inputs to machine learning models that an attacker has intentionally designed to cause the model to make a mistake; they're like optical illusions for machines. In this post we'll show how adversarial examples work across different mediums, and will discuss why securing systems against them can be difficult.

The Dark Side of Automated Domain Validation Certificate Issuance
https://www.thesslstore.com/blog/lets-encrypt-paypal/amp/
Vincent Lynch / March 6th (Yesterday) / The SSL Store
A Call To Let's Encrypt: Stop Issuing "PayPal" Certificates
Data shows wide-spread use of certificates for phishing

This is an open letter to Let's Encrypt regarding its issuance of certificates containing the word "PayPal."

Within the SSL/Certificate Authority industry, there is an ongoing debate about SSL certificates for malicious websites – the big question is if CAs should be policing the content and nature of the sites they issue certificates to. Should CAs be filtering out and rejecting certificate requests if they believe websites will use them for phishing or malware distribution? Should CAs revoke certificates for websites that are reported and proven to be involved in these activities?

Prior to Let's Encrypt, all major CAs supported the view that certificates for malicious sites should be rejected or revoked, and Let's Encrypt has stirred the pot by taking such an aggressive stance on the subject.

The Executive Director of Let's Encrypt has previously written about his view on a Certificate Authorities' responsibilities. They think it's not a CA's job to determine if the site requesting a certificate is safe or legitimate, and that even when one tries to, CAs aren't very effective at blocking the "bad" sites.

As a result, Let's Encrypt forgoes the pre-issuance checks that CAs have traditionally used to block "high-risk" requests likely to be used for malicious reasons, such as phishing. Instead, Let's Encrypt defers to services like Google's Safe Browsing and Microsoft's SmartScreen which identify and block dangerous sites at a different layer.

Most of the commercial CAs disagree with Let's Encrypts position and this is a topic that is frequently debated. For more background on this topic, I suggest reading this great post from Eric Lawrence, which inspired this post.

I am not asking Let's Encrypt to change its larger position. I respect and understand its view, and think it's a sensible position given its goals as a CA.

Given that the content filtering debate is such a heated topic, I would like to sidestep it all together and ask for something much simpler:

Stop issuing certificates containing "Paypal."

Certificates containing the term "PayPal" are being pervasively abused, and the continued issuance of these certificates poses a danger to the web by bestowing legitimacy to phishing sites. Let's Encrypt can address this without impacting its users or its mission.

That's it. That's all we're asking. Now you may be thinking, wait, isn't this content filtering?

As other CAs implement it, filtering involves complicated blacklists, submissions to multiple reputation and spam services, manual review processes, and a constant whack-a-mole game figuring out what misspelling or homonym the phishers are using this week.

There is no way for Let's Encrypt to implement similar measures without it compromising its mission or incurring large costs in developing, maintaining, and reviewing such measures. I think it is unfair to ask for that.

Instead, simply blocking "PayPal" (and literally just "PayPal," no variations or misspellings) is an easy, feasible, and effective measure against the most dangerous and malicious use of Let's Encrypt certificates.

When Eric published his post, Let's Encrypt had issued 709 certificates containing "PayPal." Now that number is 988.

988 Let's Encrypt PayPal certificates.

Eric Lawrence
https://textslashplain.com/2017/01/16/certified-malice/

By December 8, 2016, LetsEncrypt had issued 409 certificates containing "Paypal" in the hostname; that number is up to 709 as of this morning. Other targets include BankOfAmerica (14 certificates), Apple, Amazon, American Express, Chase Bank, Microsoft, Google, and many other major brands. LetsEncrypt validates only that (at one point in time) the certificate applicant can publish on the target domain. The CA also grudgingly checks with the SafeBrowsing service to see if the target domain has already been blocked as malicious, although they "disagree" that this should be their responsibility. LetsEncrypt's short position paper is worth a read; many reasonable people agree with it.