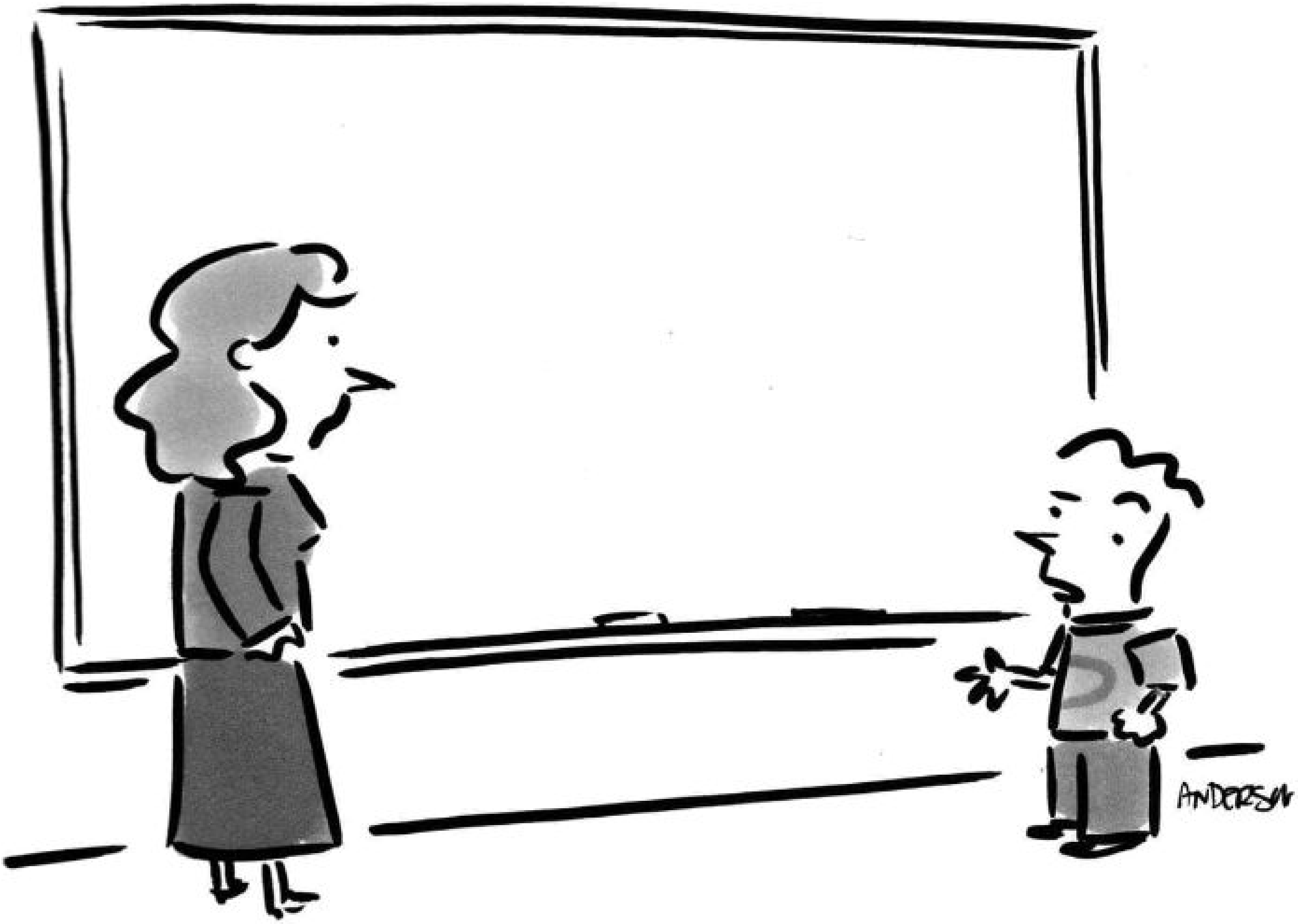
Security Now! #750 - 01-21-20

The Crypto CurveBall

This week on Security Now!

This week we look at Google's addition of iOS devices as full Google account logon hardware security keys, as update on Apple vs Attorney General Barr, a serious new Internet Explorer 0-day and how the vulnerability can be mitigated, the release of Microsoft's Chromium-based Edge browser, the FBI's reaction to the Pulse Secure VPN vulnerability, another new and CRITICAL RDP remote code execution vulnerability that has slipped under the radar, a bit of miscellany, and then we examine the the headline grabbing CryptoAPI vulnerability that's been dubbed "CurveBall."

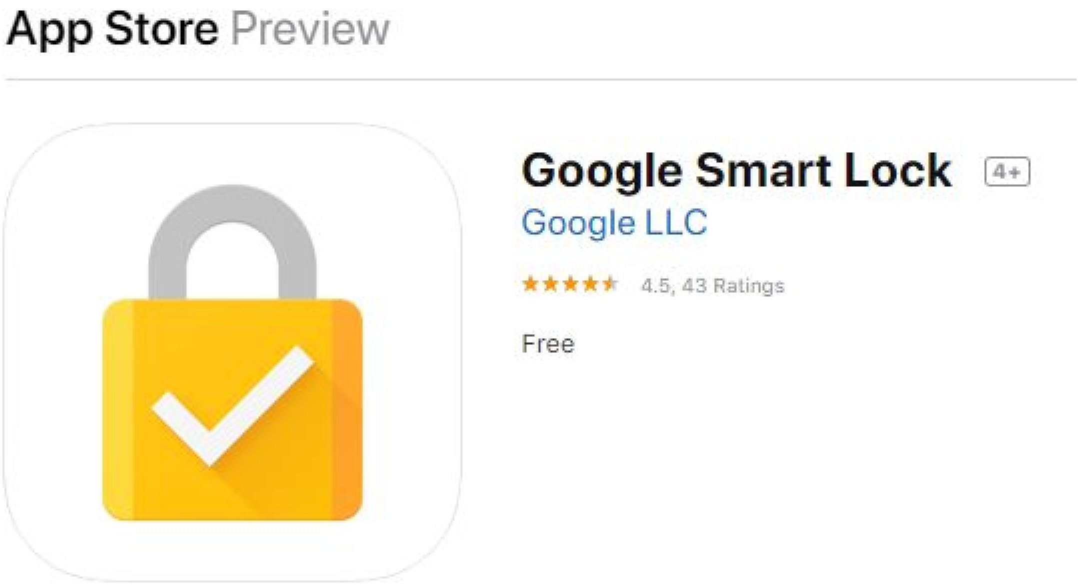


"Before I write my name on the board, I'll need to know how you're planning to use that data."

Security News

iPhones join Android in being a Google account security key.

Last Spring Google allowed Android devices to double as hardware security keys for use in providing multi-factor authentication to Google services. Now, that feature is being rolled out to users of iOS v10 and later devices.



App Store Preview

Google Smart Lock 4+

Google LLC

★★★★★ 4.5, 43 Ratings

Free

https://apps.apple.com/app/google-smart-lock/id1152066360

You first download and install the Google Smart Lock application (above). Bluetooth must be active on both your phone and on the device you wish to authenticate to -- however, the devices do not need to be paired to each other over Bluetooth.

You then need to verify that you have 2-Step Verification or Advanced Protection turned on (or turn it on) by visiting https://myaccount.google.com/security. Under "Signing in to Google," select 2-Step Verification. You'll need to sign in again to prove you're you doing this even if you're already signed in. Then scroll down to find and click "Add security key" and choose the iPhone you want to use, then "Add." Follow the instructions and enable your iPhone's security key by tapping "Yes, I'm in" when prompted to in the Smart Lock app.

From then on, you can use iPhone's security key to sign in to your Google account on new devices by:

- Verifying that Bluetooth is active on both devices.
- Sign in to your Google Account on Chrome OS (version 79 and above) or a browser on iOS, macOS or Windows.
- Check your iPhone for a Smart Lock notification. Tap the notification.
- To verify your sign-in, tap Yes.

And to make sure that your super-security doesn't also lock you out, Google recommends that you register a backup security key to your Google account to use in the event that you lose your iPhone. It wasn't clear to me what a "backup security key" means… Another phone? A traditional hardware key? Perhaps an OTP that's never used? But it's clear that you'll want to have some fallback means of getting into your Google account it you have tethered access to your iPhone and your iPhone is not around.

We talked a lot last year about the scourge of Phishing attacks which rely upon and prey upon human fallibility to perpetrate successful attacks. In some of the reporting about this I was surprised by glad to learn that Gmail blocks more than 100 Million Phishing attacks PER DAY. Wow.

And speaking of Apple…

Our listeners know that I am fascinated by the interplay between encryption and law enforcement.

As we first discussed last week, US Attorney General William Barr called upon Apple to help the FBI circumvent the password-protection used on the two iPhones which were in the possession of the gunman who is accused of shooting three Navy personnel at the Air Force base in Florida last month. William Barr stated that the shooting by Saudi air force trainee Mohammed Saeed Alshamrani was an act of terrorism and that Apple hasn't helped the FBI's investigators access data on the shooter's iPhones. At a press conference Barr stated: <quote> “We have asked Apple for their help in unlocking the shooter's iPhones. So far Apple has not given us any substantive assistance.”

I suppose that the characterization as “substantive” is somewhat open to interpretation. But based upon the level of Apple's quite substantive cooperation, Barr muse have meant that Apple didn't give them absolutely everything they asked for.

Apple replied to Barr's characterization by saying: “We reject the characterization that Apple has not provided substantive assistance in the Pensacola investigation. Our responses to their many requests since the attack have been timely, thorough and are ongoing. Within hours of the FBI's first request on December 6, we produced a wide variety of information associated with the investigation. From December 7th through the 14th, we received six additional legal requests, and in response provided information including iCloud backups, account information and transactional data for multiple accounts. We responded to each request promptly, often within hours, sharing information with FBI offices in Jacksonville, Pensacola and New York. The queries resulted in many gigabytes of information which we turned over to investigators. In every instance, we responded with all of the information we had."

Apple added it received the subpoena for the second iPhone on January 8th and responded to the FBI's request within hours of receiving it.

Barr continued by adding... “This situation perfectly illustrates why it is critical that investigators be able to get access to digital evidence once they have obtained a court order based on probable cause. We call on Apple and other technology companies to help us find a solution so that we can better protect the lives of Americans and prevent future attacks.”

So this sounds more like Barr just continuing to bang the drum for change and not actually expecting anything else from Apple. Barr said the FBI had received a warrant to search both iPhones to find out who the shooter was communicating with. And we did learn a bit more, Barr said: "During the gunfight with first-responders, the shooter disengaged long enough to place one of the phones on the floor and shoot a single round into the device. It also appears the other phone was damaged. [But] our experts at the FBI crime lab were able to fix both damaged phones so they are operational. However, both phones are engineered to make it virtually impossible to unlock them without the password. It is very important to know with whom and about what the shooter was communicating before he died."

And, as I noted last week and as we have previously covered extensively here, Apple has deliberately designed and implemented within their iDevices a robust system for encrypting the data stored within iDevice non-volatile storage. The upshot of that design is that no one, not even they, have any means or ability to decrypt it. And I'm sure they have explained this at great length in their replies. So it's unclear why Barr continues to imagine that Apple is simply resisting. We know from the design of the locking system that they are not resisting. It's clear that they cannot decrypt a user's locked device. Period.

If Bill Barr wishes to outlaw the commercial sales of such technology in the US, he'll need Congress to pass legislation to that effect. Until then he's just complaining that Apple cannot do something that was deliberately designed to be impossible.

A brand new serious Internet Explorer 0-day

Over the weekend, news dropped of a serious new IE 0-day Remote Code Execution exploit. Of biggest concern are the 1 in 4 desktops running Windows 7, since none of them will presumably ever be patched and, unless their users have switched to Chrome, Firefox or another browser, they will forever remain vulnerable. And even if another browser is setup on those systems by default, it's still possible to explicitly invoke IE to take over a machine.

https://portal.msrc.microsoft.com/en-us/security-guidance/advisory/ADV200001#ID0EWIAC

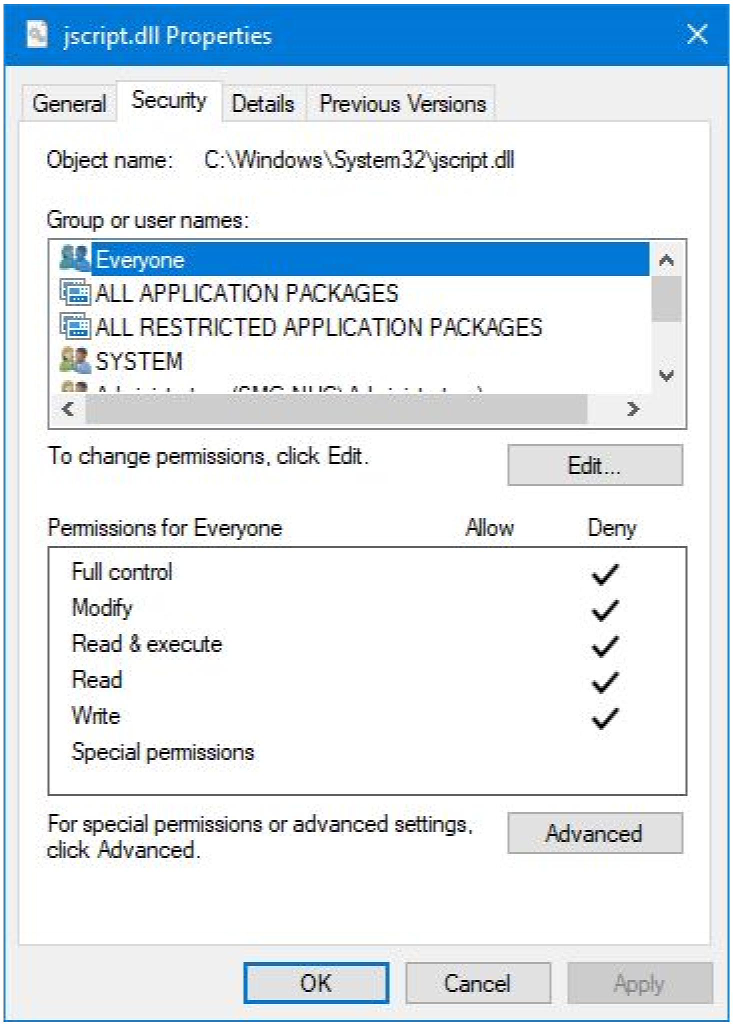
Okay... So first of all, the problem is a 0-day because it was discovered being used in targeted attacks in the wild. So this is not a theoretical issue, it's very real.

It turns out that there's a flaw in one of two available JavaScript librarys (JSCRIPT.DLL) which can be used by IE 9, 10 and 11. By default the IE browsers use JSCRIPT9.DLL which is not vulnerable. But since a malicious site or advertisement would request the use of the vulnerable JSCRIPT.DLL, this doesn't help us much. And since IE 9 through 11 exist in versions 7, 8.1 and 10, the problem affects all versions of Windows currently in use.

Of course, 8.1 and 10 =WILL= be updated, though it's unclear when. We just received our monthly Windows booster-shot, and it's unclear how serious Microsoft is taking this one. So we don't know whether we'll see an out-of-cycle emergency patch before February's patches. However, the vulnerability, which is being tracked as CVE-2020-0674 is only rated "Moderate" -- even though it is a Remote Code Execution vulnerability resulting from the way IE's JavaScript engine handles objects in memory. And as a result a remote attacker can execute arbitrary code on targeted computers and take full control of them simply by convincing victims to opening a maliciously crafted web page on the vulnerable Microsoft browser... Or, of course, by injecting a malicious advertisement into the Internet's ad stream.

Microsoft's advisory says: "The vulnerability could corrupt memory in such a way that an attacker could execute arbitrary code in the context of the current user. An attacker who successfully exploited the vulnerability could gain the same user rights as the current user." (Remember that this is Microsoft's coy voice. They found this being done in the wild, not because it could be, but because is WAS BEING.) They continue: "If the current user is logged on with administrative user rights, an attacker who successfully exploited the vulnerability could take control of an affected system. An attacker could then install programs; view, change, or delete data; or create new accounts with full user rights. Microsoft is aware of 'limited targeted attacks' in the wild and is working on a fix, but until a patch is released, affected users have been provided with workarounds and mitigation to prevent their vulnerable systems from cyberattacks.

Which brings me to my next point: The workarounds. It's possible to quickly and easily disable **all** access to the vulnerable jscript.dll… which is what I immediately did, since I never deliberately use IE and any IE-dependent sites barely legitimately use jscript.dll. So the chance of encountering any problem is next to zero.



Object name: C:\Windows\System32\jscript.dll

Group or user names:

- Everyone
- ALL APPLICATION PACKAGES
- ALL RESTRICTED APPLICATION PACKAGES
- SYSTEM

To change permissions, click Edit.

Permissions for Everyone	Allow	Deny
Full control		✓
Modify		✓
Read & execute		✓
Read		✓
Write		✓
Special permissions		

For special permissions or advanced settings, click Advanced.

The dialog above shows the properties of my Win10 system's jscript.dll after "Everyone" has been denied any rights to read or write it in any way.

Restrict access to JScript.dll

For 32-bit systems, enter the following command at an ***administrative*** command prompt:

```
takeown /f %windir%\system32\jscript.dll
cacls %windir%\system32\jscript.dll /E /P everyone:N
```

For 64-bit systems, enter the following command at an ***administrative*** command prompt:

```
takeown /f %windir%\syswow64\jscript.dll
cacls %windir%\syswow64\jscript.dll /E /P everyone:N
takeown /f %windir%\system32\jscript.dll
cacls %windir%\system32\jscript.dll /E /P everyone:N
```

How to undo the workaround

For 32-bit systems, enter the following command at an administrative command prompt:

```
cacls %windir%\system32\jscript.dll /E /R everyone
```

For 64-bit systems, enter the following command at an administrative command prompt:

```
cacls %windir%\system32\jscript.dll /E /R everyone
cacls %windir%\syswow64\jscript.dll /E /R everyone
```

Since most sites will not be requesting the use of the vulnerable JSCRIPT.DLL, disabling it until Microsoft sends us a patch seems a simple and useful precaution. Although it is currently being used only in targeted attacks, the bad guys will also be aware that a great many of the great many 1-in-4 desktops still running Windows 7 may well still be using the default IE11 browser. And since those desktops that will never be updated represents a HUGE number of PCs, we might very well see an escalation of the use of this now-persistent 0-day remote code execution vulnerability from targeted to mass spray.

Giving Windows an additional Edge

And speaking of browsers, Microsoft also last week took the wraps off of their new Chromium based Edge browser. So I guess Paul and MaryJo can now stop referring to it as "Chredge"...

https://www.microsoft.com/en-us/edge
https://grc.sc/edge

It is immediately available for download for Windows 7, 8.1, 10, iOS and Android platforms. As we know, this Chromium-based Edge browser abandons Microsoft's EdgeHTML rendering engine in favor of Google's open-sourced Chromium with its Blink rendering engine, to bring bettter compatibility and performance. This first Stable release is Edge 79 and anyone who doesn't want to wait can download it immediately using the link above. Over the coming months it will be installed automatically through Windows Update. (Except for Windows 7, of course.) Microsoft plans to first release it to Windows Insiders in the Release preview ring and then slowly expand to all other Windows 10 users. The new Edge will automatically replace the existing Edge browser. It's possible to keep "Classic Edge" (would that be Cledge?) and run both side-by-side, but it’s unclear why anyone would need or want to.

I've installed it and it's quite attractive. I won't miss the original one. But it's more than just a pretty face. Microsoft has taken the Chromium core and added some nice features. Or, at least, plans to. So, after you install it, you should fire it up and browse through its Settings. The Privacy and Services section offers three flavors of tracking prevention: "Basic", "Balanced" and "Strict". Mine’s now set to Strict.

We’re also supposed to be able to “Block potentially unwanted apps” with a setting near the bottom of that section. I've seen screen shots of this on the web... But my newly downloaded instance of Edge isn’t displaying that option. I hope it'll be showing up later.

And, of course, there’s the ever-popular "Block Media Autoplay" Thank God! The blocking is not enabled by default, but the link: edge://settings/content/mediaAutoplay will take you to the page to enable blocking the autoplaying of crap you didn't ask for when a page loads.

From our "Well, that's no surprise" department...

BleepingComputer reports that the FBI has sent a flash security alert that nation-state actors have breached the networks of a US municipal government and a US financial entity by exploiting the critical authentication bypass vulnerability in Pulse Secure VPN servers.

The FBI says that unidentified threat actors have used the Pulse Secure VPN authentication bypass flaw "to exploit notable US entities" since August of 2019. In August 2019, attackers gained access to a US financial entity’s research network by exploiting unpatched VPN servers. The same month, a US municipal government network was breached in an attack that exploited the same vulnerability. Based upon the sophistication of the Tactics, Techniques, and Procedures (TTPs) used in the two attacks, "the FBI believes unidentified nation-state actors are involved in both compromises; however, it remains unclear if these are isolated incidents."

According to the FBI, the attack that targeted and compromised the US municipal government network took place in mid-August 2019. In this attack... "the operators enumerated and exfiltrated user accounts, host configuration information, and session identifiers that could allow them to gain further access to the internal network.

In the other known attack "The intruder(s) remotely exploited a Pulse Secure VPN appliance" the flash alert says. "The vulnerability in Pulse Secure allowed directory transversal and access to a file where login credentials were written in plain text. In addition, the Pulse Secure appliance may have been vulnerable to a buffer overflow and command injection. After breaching the network, the nation-state actors gained access to the Active Directory, harvesting and exfiltrating user credentials (usernames and passwords) for the VPN client

Following attempts to enumerate and gaining access to other network segments, the hackers were only able to infiltrate the exploited segment which was the only one on the network using single-factor authentication. "The intruder(s) attempted to access several Outlook web mail accounts but were unsuccessful due to the accounts being on separate domains requiring different credentials not obtained by the intruder(s). While the intruder(s) performed additional enumeration, there was no evidence that any data was compromised or exfiltrated, and the intruder(s) seemingly did not install any persistence capability or foothold in the network."

While the FBI did not directly connect these attacks to Iranian-backed hackers, a Private Industry Notification (PIN) detailing Iranian Cyber Tactics and Techniques shared a day later mentions "information indicating Iranian cyber actors have attempted to exploit the same vulnerability." "The FBI assesses this targeting, which has occurred since late 2019, is broadly scoped and has affected numerous sectors in the United States and other countries. The FBI has observed actors using information acquired from exploiting these vulnerabilities to further access targeted networks, and establish other footholds even after the victim patched the vulnerability."

What's clear is that, today, the term "threat landscape" is not hyperbole. We don't just face disorganized opportunistic hackers operating out of their parent's basement. Cyberwarfare is a REAL thing... and there are serious players scrutinizing every announced vulnerability -- just waiting for an opportunity to gain an advantage. When something like a VPN authentication bypass is announced, the bad guys give it their rapt attention while the suckers who are using the vulnerable VPN endpoints float along in blissful ignorance.

Remember that international foreign currency exchange, Travelex, that was hit by Sodinokibi ransomware on December 3 after not patching their 7 Pulse Secure VNP servers? Travelex was one of the companies that the Bad Packets guys warned of having vulnerable servers four months earlier, back on September 13th, 2019. Travelex never replied to the warning eMail.

RDP: Once more unto the breach, dear friends, once more.
U.S. Department of Homeland Security's CISA published "Alert (AA20-014A)" titled: "Critical Vulnerabilities in Microsoft Windows Operating Systems"

https://www.us-cert.gov/ncas/alerts/aa20-014a

The serious CryptoAPI vulnerability which the NSA discovered in Win10 and Windows servers, which Microsoft rated as being merely "Important," and which has now been dubbed "CurveBall," obtained the lion's share of the attention last week. And we'll be looking at it next. But this curveball obscured an entirely different vulnerability that Microsoft rates as "Critical," for reasons we will see. Believe it or not, it is yet another "pre-authentication" remote code execution vulnerability in Microsoft's Remote Desktop Protocol.

https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2020-0610

Titled: CVE-2020-0610 | Windows Remote Desktop Gateway (RD Gateway) Remote Code Execution Vulnerability

<quote> "A remote code execution vulnerability exists in Windows Remote Desktop Gateway (RD Gateway) when an unauthenticated attacker connects to the target system using RDP and sends specially crafted requests. This vulnerability is pre-authentication and requires no user interaction. An attacker who successfully exploited this vulnerability could execute arbitrary code on the target system. An attacker could then install programs; view, change, or delete data; or

create new accounts with full user rights.

To exploit this vulnerability, an attacker would need to send a specially crafted request to the target systems RD Gateway via RDP.”

This affects Windows Servers 2012, 2012R2, 2016 and 2019.

Under "Mitigations" they say: Microsoft has not identified any mitigating factors for this vulnerability.

Under "Workarounds" they say: Microsoft has not identified any workarounds for this vulnerability.

Under FAQ: What network ports are vulnerable to this attack?
The vulnerability only affects UDP transport, which by default runs on UDP port 3391.

Back on May 14th of 2019 we first learned of the now-widely-known “Bluekeep” vulnerability. Microsoft's disclosure of "CVE-2019-0708 was titled "Remote Desktop Services Remote Code Execution Vulnerability"

And back then Microsoft's description read:

> A remote code execution vulnerability exists in Remote Desktop Services – formerly known as Terminal Services – when an unauthenticated attacker connects to the target system using RDP and sends specially crafted requests. This vulnerability is pre-authentication and requires no user interaction. An attacker who successfully exploited this vulnerability could execute arbitrary code on the target system. An attacker could then install programs; view, change, or delete data; or create new accounts with full user rights. To exploit this vulnerability, an attacker would need to send a specially crafted request to the target systems Remote Desktop Service via RDP.

Sounds familiar, doesn’t it? While this exploit is against the server-oriented Remote Desktop Gateway, which is hopefully less widely deployed, we are back here once again with a VERY serious vulnerability which will surely result in another massive round of exploitation.

KryptosLogic has examined the DLL that was repaired by this update. In their introduction they explain the intent of the Remote Desktop Gateway:

> Remote Desktop Gateway (RDG), previously known as Terminal Services Gateway, is a Windows Server component that provides routing for Remote Desktop (RDP). Rather than users connecting directly to an RDP Server, users instead connect and authenticate to the gateway. Upon successful authentication, the gateway will forward RDP traffic to an address specified by the user, essentially acting as a proxy. The idea is that only the gateway needs to be exposed to the Internet, leaving all RDP Servers safely behind the firewall. Due to the fact that RDP is a much larger attack surface, a setup properly using RDG can significantly reduce an organization’s attack surface.

https://www.kryptoslogic.com/blog/2020/01/rdp-to-rce-when-fragmentation-goes-wrong/

Hmmm. That didn’t work out so well. Kryptos Logic fully reverse-engineered the vulnerability and it’s yet another problem associated with UDP packet fragmentation and out-of-order reassembly after receipt. Fragmented UDP packet reassembly has historically been a huge headache for IP-stack implementers. It is inherently prone to mistakes. And, indeed, it bit us again.

This problem and its disclosure is only a week old. So perhaps there are some systems that will still be updated once they are restarted. But it's also clear from our many previous iterations that many systems will **never** be updated.

So I wanted to make note of this, since the more sexy "CurveBall" CryptoAPI tech press coverage tended to miss this also very serious new revelation.

SQRL News

SQRL for Drupal

Jürgen Haas: "Happy to announce that I was finally able to finish off the SQRL integration into Drupal 8 (and the forthcoming version 9). It is feature complete and supports all of the SQRL protocol version 1.

Miscellany

Revamped certificates for GRC: DigiCert "Secure Site Pro SSL" - 2 years

- EV certs cannot have wildcard domain names.
- I have a growing number of grc.com subdomains.
- Browsers no longer celebrate (or even disclose without prompting) EV certification.
- My main grc.com certificate was coming up for renewal in March.
- So… It only made sense to switch.

The process with DigiCert was as shockingly painless as every interaction has been. Since I was already approved for all of the domains I needed to bundle into the single certificate, I simply used their Windows certificate manager to generate a CSR (certificate signing request) for the root and wildcard domains I needed… and I had a deployable certificate a few minutes later.

The Crypto CurveBall

We all understand the critical importance of the Internet's trust model which, with the caveats we often examine, for the most part works. We trust that we have established a private connection to a website because of the HTTPS certificates presented by the server which is automatically verified by our browser and rejected if anything seems wrong. Our browsers are careful to warn us when we encounter a problem, such as an invalid certificate. So, we enter our credit card numbers, interact with our banks, and essentially trust in these technologies to keep us safe.

But, CVS-2020-0601 has upset the easy trust we have developed in that model. Fortunately, this is not due to any problem with the design of the PKI public key infrastructure but because the implementation of the certificate checking on some Windows systems is flawed in a way that makes it vulnerable to a failure to detect invalid, spoofed certificates.

As we know, last Tuesday Microsoft issued a security update to fix an "important" vulnerability in the Microsoft CryptoAPI (crypt32.dll) which had been discovered and privately reported by the US NSA. The fix, in short, ensures that the Windows CryptoAPI library completely and properly validates ECC certificates. Until this fix has been put in place it's possible for certificates to be deliberately manipulated in a way that fools Windows into believing that a fraudulent certificate is legitimate.

To be a bit more specific, this flaw, now referred to as "CurveBall", was introduced into Windows' code in July of 2015. It's a spoofing vulnerability in the way the certificates are accepted without a proper verification of the explicit curve parameters within the certificates. Essentially, this flaw allows an attacker to supply his own generated X.509 certificates by using an “explicit parameters” option to set those curve parameters.

The NSA wrote: “Certificates containing explicitly-defined elliptic curve parameters which only partially match a standard curve are suspicious, especially if they include the public key for a trusted certificate, and may represent bona fide exploitation attempts.”

So, the way for us to think about this is that Microsoft's original code was allowing certificates to over-specify their own elliptic curve parameters, which were then used to verify their own signatures. The certificate said both: "This is how you verify who I am" and "This is who I am." Whoops.

So this flaw affects all of the places where we depend upon trusting certificates, including HTTPS connections, signed files and emails, and signed executable code.

What this means from an adversarial context is that by leveraging Curveball, X.509 certificate chain validation is susceptible to:

MITM Attacks, Signed File/Malware Attacks, and anywhere we rely upon something having been signed by some trusted authority. It's a full signature bypass affecting the codebases of Windows 10, Windows Server 2016 and Server 2019. Although Windows 7 and Server 2008R2 also received (their final) updates last week, their Crypto32.dll's were never vulnerable.

https://media.defense.gov/2020/Jan/14/2002234275/-1/-1/0/CSA-WINDOWS-10-CRYPT-LIB-20190114.PDF

And, moreover, unlike the famous Specter and Meltdown vulnerabilities, which remained essentially theoretical, we already have multiple working proofs-of-concept.

For the past week many researchers have been rushing to create and release proof-of-concept code to demonstrate that the vulnerability can indeed be exploited in the wild. Researchers predicted that it may only be a matter of days and this prediction turned out to be spot-on. The first researcher to prove the vulnerability was exploitable was Saleem Rashid who developed a PoC code that permitted the faking of TLS certificates.

As we know, faking a TLS certificate allow hackers to create fake websites that not only look like the real deal but will also have a certificate that says it's legitimate. Rashid hasn't shared his code, but he tweeted some visual proof:

https://twitter.com/saleemrash1d/status/1217495681230954506

The published code came from Kudelski Security and it was followed closely by a Danish researcher Ollypwn. Both now have proofs-of-concept published on Github:

Kudelski calls his “Chain Of Fools”: https://github.com/kudelskisecurity/chainoffools

We have a full explainer and complete proof-of-concept code:
https://research.kudelskisecurity.com/2020/01/15/cve-2020-0601-the-chainoffools-attack-explained-with-poc/

OllyPwn: https://github.com/ollypwn/cve-2020-0601

> CVE-2020-0601, or commonly referred to as CurveBall, is a vulnerability in which the signature of certificates using elliptic curve cryptography (ECC) is not correctly verified.
>
> ECC relies on different parameters. These parameters are standardized for many curves. However, Microsoft didn't check all these parameters. The parameter `G` (the generator) was not checked, and the attacker can therefore supply his own generator, such that when Microsoft tries to validate the certificate against a trusted CA, it'll only look for matching public keys, and then use then use the generator of the certificate.
>
> `MicrosoftECCProductRootCertificateAuthority.cer` is by default a trusted root certificate authority (CA) using ECC on Windows 10. **Anything** signed with this certificate will therefore automatically be trusted.

So now we have publically available code and exploitation in the wild by malicious actors will be next. It's worth nothing that Google immediately responded by updating Chrome to its current v79.0.3945.130 which blocks this exploit in the browser. Saleem's tweet clearly indicated that

Firefox was always safe. Firefox brings along its own NSS security suite which was never vulnerable.

It's understandable why this commanded so much attention. It fully undermines the entire Windows certificate trust store and facility... upon which we have come to depend. And, unlike some exploits that take real skillz, this one is a no-brainer and exploits appeared almost instantly. So there is NO CHANCE that bad guys won't be attempting to exploit this against any Windows 10 desktops and servers that haven't yet been updated in the past five years.