

# Security Now! #776 - 07-21-20

## A Tale of Two Counterfeits

### Mea culpa

After last week's podcast, during which I rather clearly expressed my lack of understanding of some people's sensitivity to terminology, a number of our listeners reached out to share their distress and disappointment with me. I responded to each and every one of the people who wrote and we engaged in very productive dialog. I am embarrassed in the presence of their grace, and I sincerely apologize to those listeners whom I offended with my rant, which belittled something that they are sensitive to. That's not my place. This is not to say that I fully appreciate the sensitivity to terminology that clearly has nothing to do with racial prejudice. But thanks to the conversations I had with some of our listeners, the way I CAN understand this is by seeing this change in terminology for what it REPRESENTS: It is a proxy for some clearly very much needed progress toward the recognition of the full equality that we are each obviously entitled to by birth. So, I get it. And the other thing I get is that NO ONE is tuning in to this podcast for an update on my feelings about social justice. I am SO impressed by the quality of our listeners. I thank you for putting up with me. Now let's talk about stuff I DO understand...

### This week on Security Now!

This week we, of course, start off by looking at what happened at Twitter last week. We look at CheckPoint's discovery of the headline grabbing wormable DNS vulnerability that's been present in all Windows Servers for the past 17 years, we touch on last week's Patch Tuesday, look at Cloudflare's surprise outage, another glitch in Zoom's product, and 7 "no-logging" VPN providers whose logs were all found online. We cover some other quick news, some interesting SpinRite development developments, then we examine the problem of counterfeit networking equipment which, as our picture of the week shows, is actually a big problem...



**Which one is the fake??**

# Security News

## The Twitter Incident

As probably everyone who routinely uses the Internet knows by now, last Wednesday Twitter was hacked. There are any number of summaries on the Net of what happened. I've read all I could find. But none seem as clear to me as Twitter's own summary which they shared last Friday night after two and a half days of frantic forensics:

[https://blog.twitter.com/en\\_us/topics/company/2020/an-update-on-our-security-incident.html](https://blog.twitter.com/en_us/topics/company/2020/an-update-on-our-security-incident.html)

They titled it: "An update on our security incident" and they explained:

In this post we summarize the situation as of July 17 at 8:35p Pacific Time. The following information is what we know as of today and may change as our investigation and outside investigations continue. Additionally, as the investigation of this incident is unfolding, there are some details — particularly around remediation — that we are not providing right now to protect the security of the effort. We will provide more details, where possible in the future, so that the community and our peers may learn and benefit from what happened.

### What happened

At this time, we believe attackers targeted certain Twitter employees through a social engineering scheme. What does this mean? In this context, social engineering is the intentional manipulation of people into performing certain actions and divulging confidential information.

The attackers successfully manipulated a small number of employees and used their credentials to access Twitter's internal systems, including getting through our two-factor protections. As of now, we know that they accessed tools only available to our internal support teams to target 130 Twitter accounts.

*[I'll note that they are quite sensitive about those tools. Motherboard obtained some screenshots which appeared authentic. But when someone tweeted them, those tweets were removed and the poster's account was suspended. I tweeted a link to the Motherboard article in case anyone's interested.]*

For 45 of those [targeted 130] accounts, the attackers were able to initiate a password reset, login to the account, and send Tweets. We are continuing our forensic review of all of the accounts to confirm all actions that may have been taken. In addition, we believe they may have attempted to sell some of the usernames.

For up to eight of the Twitter accounts involved, the attackers took the additional step of downloading the account's information through our "Your Twitter Data" tool. This is a tool that is meant to provide an account owner with a summary of their Twitter account details and activity. We are reaching out directly to any account owner where we know this to be true. None of the eight were verified accounts.

## **Our actions**

We became aware of the attackers' action on Wednesday, and moved quickly to lock down and regain control of the compromised accounts. Our incident response team secured and revoked access to internal systems to prevent the attackers from further accessing our systems or the individual accounts. As mentioned above, we are deliberately limiting the detail we share on our remediation steps at this time to protect their effectiveness and will provide more technical details, where possible, in the future.

In addition to our efforts behind the scenes, shortly after we became aware of the ongoing situation, we took preemptive measures to restrict functionality for many accounts on Twitter - this included things like preventing them from Tweeting or changing passwords. We did this to prevent the attackers from further spreading their scam as well as to prevent them from being able to take control of any additional accounts while we were investigating. We also locked accounts where a password had been recently changed out of an abundance of caution. Late on Wednesday, we were able to return Tweeting functionality to many accounts, and as of today, have restored most of the accounts that were locked pending password changes for their owners.

We are continuing our investigation of this incident, working with law enforcement, and determining longer-term actions we should take to improve the security of our systems. We have multiple teams working around the clock focused on this and on keeping the people who use Twitter safe and informed.

## **What the attackers accessed**

The most important question for people who use Twitter is likely — did the attackers see any of my private information? For the vast majority of people, we believe the answer is, no. For the 130 accounts that were targeted, here is what we know as of today.

- Attackers were not able to view previous account passwords, as those are not stored in plain text or available through the tools used in the attack.
- Attackers were able to view personal information including email addresses and phone numbers, which are displayed to some users of our internal support tools.
- In cases where an account was taken over by the attacker, they may have been able to view additional information. Our forensic investigation of these activities is still ongoing.

We are actively working on communicating directly with the account-holders that were impacted.

## **Our next steps**

As we head into the weekend and next week, we are focused on these core objectives:

- Restoring access for all account owners who may still be locked out as a result of our remediation efforts.
- Continuing our investigation of the incident and our cooperation with law enforcement.

- Further securing our systems to prevent future attacks.
- Rolling out additional company-wide training to guard against social engineering tactics to supplement the training employees receive during onboarding and ongoing phishing exercises throughout the year.

Through all of this, we also begin the long work of rebuilding trust with the people who use and depend on Twitter.

We're acutely aware of our responsibilities to the people who use our service and to society more generally. We're embarrassed, we're disappointed, and more than anything, we're sorry. We know that we must work to regain your trust, and we will support all efforts to bring the perpetrators to justice. We hope that our openness and transparency throughout this process, and the steps and work we will take to safeguard against other attacks in the future, will be the start of making this right.

Notice that we're glaringly missing is any mention of WHO did this. I wonder how much Twitter knows about that and what law enforcement has in process right now? The other thing that's been observed is that obtaining this sort of access to Twitter has massive potential for abuse, yet what the attackers did was post some bitcoin scam... which seems sort of a waste for the power they briefly held and for the consequences to them if they can be found.

### **SigRed: "This is not just another vulnerability." (CheckPoint Research)**

This month's big scary wormable vulnerability turns out to have been present in Windows Server versions since Windows Server 2003 and it's been present in all subsequent versions of Windows Server through and including the most recent Server 2019. So, without knowing it, we've been living with this in our midst for the past 17 years.

Its discoverer was CheckPoint Research who named it "SIGRed". It's been assigned CVE-2020-1350 and is wormable, meaning that it can propagate among any and all Windows Servers with exposed DNS services. It's triggered by a specially crafted DNS response and since Windows Server's DNS runs with elevated SYSTEM privilege, if it's exploited successfully an attacker obtains full Domain Admin rights, effectively compromising the entire corporate infrastructure.

The way CheckPoint explained their discovery was interesting. They wrote:

"Our main goal was to find a vulnerability that would let an attacker compromise a Windows Domain environment, preferably unauthenticated. There is a lot of related research by various independent security researchers as well as those sponsored by nation-states. Most of the published and publicly available materials and exploits focus on Microsoft's implementation of SMB (EternalBlue) and RDP (BlueKeep) protocols, as these targets affect both servers and endpoints. To obtain Domain Admin privileges, a straightforward approach is to directly exploit the Domain Controller. Therefore, we decided to focus our research on a less publicly explored attack surface that exists primarily on Windows Server and Domain Controllers: WinDNS."

<https://research.checkpoint.com/2020/resolving-your-way-into-domain-admin-exploiting-a-17-year-old-bug-in-windows-dns-servers/>

A link to CheckPoint's detailed write-up is in the show notes for anyone who's interested. It's very detailed and wonderful and takes us step-by-step through their process. But I'll just hit the high points here: For every query type that a DNS server makes there's a corresponding reply. What the CheckPoint guys found was a classic math result variable sizing mistake in the parsing logic for the reply to a "SIG" (as in signature) query which is part of DNSSEC.

They discovered a mishandling of values between the 16-bit fields used by the DNS protocol and the 64-bit register math used by the code compiler. Coders know that if a 64-bit value is calculated to allocate memory, and if the result is larger than 65,535 (the maximum absolute quantity that can be represented by 16-bits), then the least 16-bits of that value will be a small integer — which is the amount of the overflow over 65,535. And if THAT smaller integer 16-bit value was then used to allocate memory for a buffer, the resulting buffer will be much too small to hold the larger calculated amount of data. And, of course, that's exactly what happened.

They discovered that by sending a DNS response containing a larger-than-64KB SIG record, they could cause a controlled heap-based buffer overflow of roughly 64KB more than a small allocated buffer. For hackers that's the golden keys to the server kingdom.

CheckPoint concluded their write by noting:

This high-severity vulnerability was acknowledged by Microsoft and was assigned CVE-2020-1350.

We believe that the likelihood of this vulnerability being exploited is high, as we internally found all of the primitives required to exploit this bug. Due to time constraints, we did not continue to pursue the exploitation of the bug (which includes chaining together all of the exploitation primitives), but we do believe that a determined attacker will be able to exploit it. Successful exploitation of this vulnerability would have a severe impact, as you can often find unpatched Windows Domain environments, especially Domain Controllers. In addition, some Internet Service Providers (ISPs) may even have set up their public DNS servers as WinDNS.

We strongly recommend users to patch their affected Windows DNS Servers in order to prevent the exploitation of this vulnerability.

As a temporary workaround, until the patch is applied, we suggest setting the maximum length of a DNS message (over TCP) to 0xFF00, which should eliminate the vulnerability. You can do so by executing the following commands:

```
reg add "HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\DNS\Parameters" /v "TcpReceivePacketSize" /t REG_DWORD /d 0xFF00 /f
net stop DNS && net start DNS
```

This is all only going public now because CheckPoint disclosed responsibly and Microsoft's July patch Tuesday, last week, fixed this bad boy.

### **And speaking of last week's July Patch Tuesday...**

This month, we received updated code to remove 123 security flaws from 13 products. Happily, none of the security bugs fixed this month have been observed being exploited in the real world, though a bunch had been shown to enable remote code execution.

Although the most worrisome was the DNS server bug we just discussed. Scrolling down through the list of this month's fixed 123 bugs is really quite sobering. We've seen the damage that can be done by elevation of privilege bugs, since today's operating systems are hosting so much content from so many disparate sources, maintaining isolation and control among them is one of any modern operating system's top jobs. This month I didn't even attempt to count those problems that were just fixed. But I decided to parse the list for the even worse Remote Code Execution vulnerabilities that were just eliminated last week. And we have...

1. .NET Framework Remote Code Execution Vulnerability
2. SharePoint Server Remote Code Execution Vulnerability
3. Windows Font Driver Host Remote Code Execution Vulnerability
4. Windows Font Library Remote Code Execution Vulnerability
5. Microsoft Graphics Components Remote Code Execution Vulnerability
6. Microsoft Graphics Remote Code Execution Vulnerability
7. Jet Database Engine Remote Code Execution Vulnerability
8. Jet Database Engine Remote Code Execution Vulnerability
9. Jet Database Engine Remote Code Execution Vulnerability
10. Microsoft Outlook Remote Code Execution Vulnerability
11. PerformancePoint Services Remote Code Execution Vulnerability
12. Microsoft Excel Remote Code Execution Vulnerability
13. Microsoft Office Remote Code Execution Vulnerability
14. Microsoft Project Remote Code Execution Vulnerability
15. Microsoft Word Remote Code Execution Vulnerability
16. Microsoft Word Remote Code Execution Vulnerability
17. VBScript Remote Code Execution Vulnerability
18. Visual Studio Remote Code Execution Vulnerability
19. Windows Address Book Remote Code Execution Vulnerability
20. Remote Desktop Client Remote Code Execution Vulnerability
21. LNK Remote Code Execution Vulnerability
22. Windows DNS Server Remote Code Execution Vulnerability
23. Visual Studio and Visual Studio Code Elevation of Privilege Vulnerability
24. Visual Studio Code ESLint Extension Remote Code Execution Vulnerability
25. Hyper-V RemoteFX vGPU Remote Code Execution Vulnerability
26. Hyper-V RemoteFX vGPU Remote Code Execution Vulnerability
27. Hyper-V RemoteFX vGPU Remote Code Execution Vulnerability
28. Hyper-V RemoteFX vGPU Remote Code Execution Vulnerability
29. Hyper-V RemoteFX vGPU Remote Code Execution Vulnerability
30. Hyper-V RemoteFX vGPU Remote Code Execution Vulnerability

This is just a daunting list of serious vulnerabilities to be fixing every four weeks. And we've been seeing this all year. This is not a brief anomaly. It's the way it is now. On Windows Weekly I've been hearing Paul and MaryJo lamenting what sure does appear to be the collapsing state of affairs with Windows—so I know I'm not an outlier here.

Just get a load of this bit of explanation, from a page on BleepingComputer posted last Friday with the headline: "*Microsoft fixed an issue where the Disk Cleanup maintenance utility could cause boot failures when launching automatically after installing Windows 10, version 2004 Build 19041.21.*" BleepingComputer wrote:

"To prevent this issue, Microsoft is using an automated troubleshooter — instead of applying an update block — to prevent Disk Cleanup from launching on its own and causing boot issues until the users install the Windows version 19041.84 update which comes with a fix for this bug. Microsoft says "This troubleshooter automatically runs twice. It runs for the first time on all devices on Windows version 19041.21. It then runs again after devices are upgraded to Windows version 19041.84. This troubleshooter cannot be run manually."

To see if the troubleshooter has launched on your device you have to check recommended troubleshooting history by going to **Start > Settings > Update & Security > Troubleshoot > View troubleshooting history**.

What?!?!? "To prevent this issue, Microsoft is using an automated troubleshooter instead of applying an update block to prevent Disk Cleanup from launching on its own and causing boot issues." What Windows has become would even confuse Rube Goldberg!

And as a developer, I can totally sympathize with the impossible task Microsoft has undertaken. Pick any one of those patched vulnerabilities fixed last week. Microsoft Office for example. It has a Remote Code Execution vulnerability? Of course it does. And not only one. We'll almost certainly have another couple fixed next month and some more the month after. Office alone has grown so massive and sprawling that it can't help but to have hundreds of still-unknown exploitable bugs.

How did this happen? It's really very simple. Microsoft decided, quite rationally, that we wanted features more than we wanted security. And I can't say they were wrong. Features are visible, bugs are hidden. Ever purchase a used car with a fresh paint job? That's what Windows gets every few years. The same increasingly old and increasingly creaky operating system, with a fresh paint job. Windows has become the used car of operating systems. But it's the one that most of us are driving. Beep beep!

## The Cloudflare Outage

It was just one single measly little line in a router's config file and suddenly Riot, Gitlab, Patreon, Authy, Medium, Digital Ocean and countless others, including, ironically, "DownDetector" became unreachable and dropped off the Net. So, last Friday evening, while Twitter was updating the world about its massive hack, John Graham-Cumming was describing what happened at Cloudflare: <https://blog.cloudflare.com/cloudflare-outage-on-july-17-2020/>

Today a configuration error in our backbone network caused an outage for Internet properties and Cloudflare services that lasted 27 minutes. We saw traffic drop by about 50% across our network. Because of the architecture of our backbone this outage didn't affect the entire Cloudflare network and was localized to certain geographies.

The outage occurred because, while working on an unrelated issue with a segment of the backbone from Newark to Chicago, our network engineering team updated the configuration on a router in Atlanta to alleviate congestion. This configuration contained an error that caused all traffic across our backbone to be sent to Atlanta. This quickly overwhelmed the Atlanta router and caused Cloudflare network locations connected to the backbone to fail.

The affected locations were San Jose, Dallas, Seattle, Los Angeles, Chicago, Washington, DC, Richmond, Newark, Atlanta, London, Amsterdam, Frankfurt, Paris, Stockholm, Moscow, St. Petersburg, São Paulo, Curitiba, and Porto Alegre. Other locations continued to operate normally.

This was not caused by an attack or breach of any kind. We're sorry for this outage and have already made a global change to the backbone configuration that will prevent it from being able to occur again.

John also provided some interesting technical detail about their setup:

Cloudflare operates a backbone between many of our data centers around the world. The backbone is a series of private lines between our data centers that we use for faster and more reliable paths between them. These links allow us to carry traffic between different data centers, without going over the public Internet.

We use this, for example, to reach a website origin server sitting in New York, carrying requests over our private backbone to both San Jose, California, as far as Frankfurt or São Paulo. This additional option to avoid the public Internet allows a higher quality of service, as the private network can be used to avoid Internet congestion points. With the backbone, we have far greater control over where and how to route Internet requests and traffic than the public Internet provides.

We've recently talked about BGP routing mistakes. They're easy to make and when a mistake is made by an organization the size of Cloudflare, they're hard to miss. This was exactly that. There was some backbone congestion in their Atlanta Georgia data center. So the team decided to remove some of Atlanta's traffic by rerouting it to other data centers on the backbone. Essentially, removing some routes. But an unappreciated comparison in routing preference levels resulted in an unanticipated result: Instead of removing the Atlanta routes from the backbone, the mistake caused the Atlanta router to start leaking all BGP routes INTO the backbone. Atlanta was inadvertently advertising itself as the proper destination for all of Cloudflare's backbone traffic. The other routers at the other data centers that received Atlanta's bold come hither shrugged and said "okay" ... and Atlanta was immediately buried and it collapsed. As John put it: "With the routes sent out, Atlanta started attracting traffic from across the backbone."

They were very embarrassed and apologetic (there seems to be a lot of that going around lately) and they have already put new safeguards in place so that nothing like that can happen again.

### **Zoom's vanity domain flaw**

Zoom recently repaired another glitch which was discovered by CheckPoint Research. CheckPoint responsibly and privately reported it to Zoom, and it was quickly fixed. After it was fixed the world learned of what had once been a problem. I phrase it this way because I continue to see that the tech press appears to have fallen in love with the term 0-day. Now, everything is a 0-day vulnerability, even when it's not. **This** incident was being called a 0-day vulnerability. It never was. Those researchers we talked about last week, who disclosed severe vulnerabilities



in C-Data's equipment, **created** multiple 0-day vulnerabilities by irresponsibly going public with their disclosure and not giving the manufacturer any opportunity to respond, fix the trouble and push patches. But CheckPoint and Zoom, working behind the scenes to fix a problem before it became publicly known, is not a 0-day. Vulnerabilities are not 0-days when they are fixed before they are publicly revealed.

<https://support.zoom.us/hc/en-us/articles/215062646-Guidelines-for-Vanity-URL-Requests>

Anyway, it turns out that Zoom allows the use of so-called Vanity URLs (they actually call it that), which are sub-domains under its primary "zoom.us" domain. CheckPoint discovered that due to improper account validation, any meeting ID could have been launched using any organization's Vanity URL, even if a meeting was set up by a separate individual account which had no relationship to the organization. It's unclear from CheckPoint's disclosure whether there was actually any subdomain validation, though I assume there must have been some.

<https://blog.checkpoint.com/2020/07/16/fixing-the-zoom-vanity-clause-check-point-and-zoom-collaborate-to-fix-vanity-url-issue/>

CheckPoint wrote: "Upon setting up a meeting, an attacker could change the invitation link URL to include any registered sub-domain. For instance, if the original invitation link was <https://zoom.us/j/7470812100>, the attacker could change it to <https://<organization's name>.zoom.us/j/7470812100>. A victim receiving such an invitation would have had no way of knowing the invitation did not actually come from the actual organization.

Anyway, it's unclear whether it was that easy (I hope not) and since it's fixed it didn't merit further digging. But I noted that CheckPoint added an interesting statistic factoid at the end of their disclosure writeup. They wrote:

It's worth noting that 90% of cyber-attacks start with a phishing email. To make sure you're doing enough to protect your organization's attack vectors, we suggest that you read the whitepaper "Humans are Your Weakest Link" to discover the daily risk posed by phishing emails.

<https://pages.checkpoint.com/phishing-attacks-put-your-business-at-risk.html>

- |                |        |
|----------------|--------|
| 1. Zoom        | 35.87% |
| 2. GoToWebinar | 22.44% |
| 3. Cisco Webex | 17.18% |

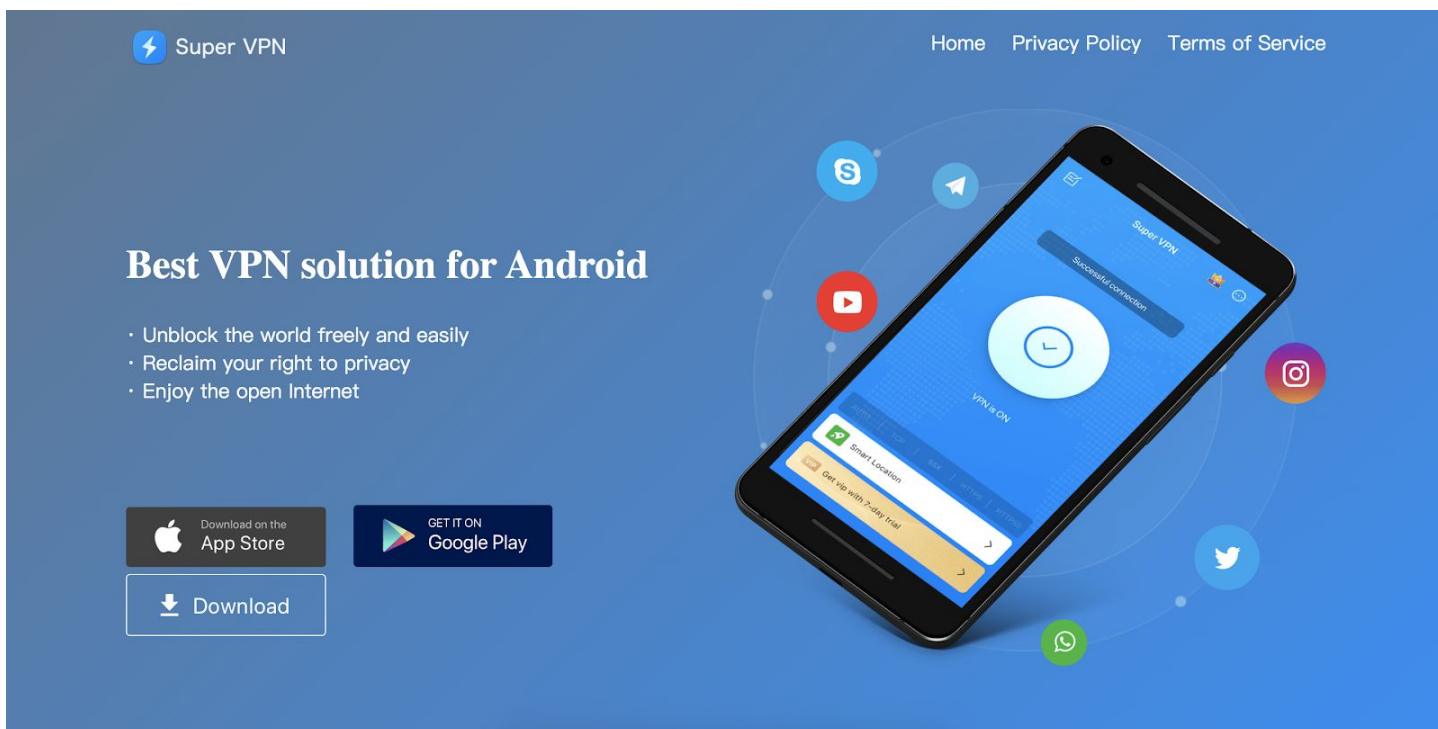
### **Not all VPNs are created equal.**

We learn this week some specific details of something we probably always suspected: It really does matter which VPN provider one chooses. "VPNmentor" is a site which obtains its revenue from affiliate links to well known VPN providers, one of them a current sponsor of the TWIT network.

Recently, the user connection and activity logs of 7, apparently different, free VPN providers who all boasted about their “zero logging” services, were discovered on the Internet, in the cloud, on an elasticsearch database instance.

The vpnMentor research team, led by Noam Rotem, discovered the database containing a staggering 1 billion database entries associated with approximately 20 millions users, so an average of 50 log entries per user... despite the fact that each of the VPN services advertises that they are “no-log” VPNs. The database contained internet activity logs with personally identifiable data including email addresses, clear text passwords, IP addresses, home addresses, phone models, device ID, and other technical details.

The good news is, these are not mainstream VPNs. Their names are UFO VPN, FAST VPN, Free VPN, Super VPN, Flash VPN, Secure VPN, and Rabbit VPN. And, while they all have separate names, they all appear to be connected to a common app developer who white-labeled one product for multiple companies. But from the outside, the typical consumer would have absolutely no way of knowing...



**Super VPN** – “Super VPN is the best unlimited VPN proxy for android.”

Google Play Store: 4.6 stars, 1M+ downloads

Apple App Store: 4.9 stars

Developer: Nownetmobi, Hong Kong

Any unwitting user might be forgiven for thinking “Hey, looks Great! A free VPN service sounds perfect!” We know that today a VPN service need not be expensive. But if actual privacy and security is one's goal, I'd rather pay a fair price and have that actually provided. It clearly does matter whom one chooses.

## **Apple updated it iOS and macOS with a handful of useful security patches.**

I had to ask for them on my iPhones. So anyone who is interested might wish to check for and solicit the update from Apple.

## **“Firefox Send” is still not receiving...**

This is becoming a curious outage, because you wouldn't think that requiring everyone to have an account tied to a verified eMail address, when that was already an option, nor adding a “Report Abuse” button, would be a heavy lift for Mozilla. So it feels as though perhaps something more substantial might be going on behind the scenes. And if they're able to make Firefox Send better any more resistant to abuse in other ways so that it doesn't again fall victim to malware purveyors, it's probably worth waiting for.

## **Listener Feedback**

Joe Lyon | @Joe\_Lyon\_

“I just had a Spinrite save with my Dell E7450 Win7 Pro laptop. I recently booted the machine and all my desktop icons, SQRL login, etc. were gone. I did a search online and it seems to be related to a corrupted profile in Windows. There were detailed steps to take to recover my profile and try and get all of the information back. Before taking all those steps and doing all that work I thought I'd try Spinrite. I booted the machine with FreeDOS and ran Spinrite on level 2. After about an hour I rebooted into windows and voilà it booted properly and all my files, icons, programs were back where they should be! Thank you Spinrite and Steve!”

## **SpinRite**

I have so much news on the SpinRite R&D front that I hardly know where to begin.

For one thing, our work so far has suggested that there may be FAR more that can be done with solid state storage than I was expecting. The timing results we're seeing from the benchmarks suggest that there's far more going on under the hood than we might first expect from solid state storage. But on second thought... of course the SSD engineers would be squeezing every last possible bit (literally) out of the technology that they're able to, just as was done with hard drives, which is what resulted in so much recovery margin. So I'm beginning to better understand why SpinRite has had so many reports of success with the recovery of data from solid state storage. I'm thinking that we'll eventually have a solid state storage assessment tool unlike anything that's been done before.

In other news, there's a time saving approach that SpinRite will be able to use for any mass storage media, spinning or solid state, when the actual transfer rate from the storage medium exceeds the transfer rate of its link to the system. That won't be an issue with NVMe storage, but it will when a fast spinning drive capable of more than 300 MB/sec is stuck behind a SATA II link that maxes out at 300 MB/sec. Or with any fast SSD that can read faster than SATA III's 600 MB/sec. It's possible to ask a drive to verify that it's able to read from its physical media without actually bothering to transfer the data across the serial SATA link. We have found that some drives apparently cheat and immediately say “yeah, no problem.” So SpinRite will be

carefully testing drives by deliberately inducing an error then checking to see whether the drive says “no problem” or “whoops.” The reason this is so exciting, when it can be done, is that for drives that faithfully honor the “Read Verify” command, SpinRite will potentially be able to perform a full media test at a very practical speed.

One of the people testing the R&D code has a 500GB Samsung 850 EVO SSD. The fastest we’re able to read from it using native command queuing is 528 MB/sec. That’s not bad. But the Samsung SSD’s properly honor the Read Verify and do the real work. So SpinRite will be able to scan any of those drives at more than 800 MB/sec. (We measured 806.) Which means that the entire 500GB drive can be media-scanned in less than 10.5 minutes.

```
> 531.657 2 cmd slot ping-pong read
> 520.047 32 preloaded AHCI cmd slots
> 528.430 32 NCQ commands sent to drive
> 806.096 2 cmd slot ping-pong verify
> 705.036 32 preloaded AHCI verify
```

Anyway, we’re making great progress. We’re currently tracking down obscure but reproducible behavior that only appears to affect HP Desktops with their BIOS in a particular setting. Working out a work around is my plan for this evening. We have a terrific group of very patient testers and we’re all having a good time nailing down the operation of this code for the next SpinRite.

---

## A tale of two counterfeits

Shortly after the surprise publication of Pierre Kim's and Alexandre Torres' findings of 7 severe and apparently deliberate egregious vulnerabilities — including hard-coded username and password backdoors for the device's Telnet server — C-Data, the manufacturer whose name was on those devices, posted a statement on their website: <https://cdatatec.com/statement/>

### Statement on Pierre Kim Revealing Security Vulnerabilities in C-Data OLT products

C-Data noticed that Pierre Kim released security vulnerabilities in C-Data OLT on the Github website. C-Data immediately started investigation and analysis. We will give report as soon as possible.

C-Data adheres to protecting the ultimate interests of users with best effort and provide customer with safety products.

Here, we express our appreciation for Pierre Kim's concern on C-Data products.

Below that opening statement was later posted a link to their full technical statement: <https://cdatatec.com/technical-statement-2/>

## Statement on Pierre Kim Revealing Security Vulnerabilities in C-data OLT products

We have noticed an article named "Multiple vulnerabilities found in C-Data OLTs" published in Github. C-Data admires the work of two professionals in technological circles, Pierre Kim and Alexandre Torres, and thanks for their identifying security breach problems through detailed testing, as well as for their active work in reducing the risks of users using network products. C-Data adheres to the philosophy of serving customers, and always puts customers' interests in the first place, as well as pays special attention to the product safety problems. In this way, C-Data can provide customers with products with safety guarantee.

In the meantime, we have paid attention to some press releases published by the media, and have interpreted technical articles by Pierre Kim and Alexandre Torres. In order not to let the majority of customers misunderstand the safety design of our equipment, C-Data analyzes and clarifies the mentioned technical issues with a sincere and frank manner.

### **Excluding counterfeit products**

The [login] account mentioned in this article: panger123/suma123. We have investigated the account and the password. In addition, we have confirmed that the account and password are not from the C-Data OLT products, but are those used by other companies and people when they copy the C-Data OLT [hardware]. The CLI style and most of its commands of the counterfeited OLT are all copied from the C-Data OLT. C-Data OLT equipment is now widely used around the world, and counterfeiters copy C-Data OLT for illegal profits.

According to the following screenshot, we can completely compare and analyze that the account of panger123/suma123 comes from an illegally copied OLT.

[And they provide two screen shots demonstrating the differences between the two.]

### **Introduction to several factory setting accounts**

The following two telnet login accounts and passwords mentioned in this article are actually used on the C-Data's first generation OLT (OLT starting with FD11XX):

**OLT telnet account 1: debug/debug124**

**OLT telnet account 2: root/root126**

This account and password are mainly used by C-Data to assist customers in debugging problems and writing production parameters. (OLT MAC address information and Serial Number information, etc.)

This account must be successfully logged in to the CONSOLE port by a local serial line [plugged into] the OLT, then can entering the OLT bcm-shell mode to modify and view key information of the OLT. Using this account under OLT TELENT mode, we can only enter the CLI of the device, we cannot enter OLT bcm-shell to modify any information of OLT.

[In other words, those telnet username and password combinations can only be used when connecting to the device's hardwired physical serial port, and even then only allow for viewing of

the device's fixed information such as its network interface MAC addresses and its serial number.]

If attacks want to enter the bcm-shell mode of OLT to obtain device privacy information or implant malicious programs into OLT, they must log into OLT by directly connecting the serial port line to the computer locally. In this way, by no means can remote attackers use these two accounts to attack. Therefore, there is no such situation as "Backdoor Access with telnet".

```
*****
Command Line Interface for EPON System
Hardware Ver: V3.1
Software Ver: 2.4.05_000
Created Time: May 17 2018 11:16:52
Copyright (c) 2006-2015 All rights reserved.
*****
Username: root
Password:
epon# bc

epon# debug bcm-shell
Only the console support!
epon# █
```

### OLT telnet Account3 : guest/[empty]

The account and password are the account of factory default configuration, which can only check some basic information of OLT, and without having the authority to configure any OLT. The user can delete or modify the account as needed when using it.

### More Secure Cryptographic Mechanism

For other models of C-Data OLTs(OLT named FD15XX, FD16XX, FD12XX, FD8000), the problem of "Backdoor Access with telnet" does not exist, because these OLTs adopt a more secure cryptographic mechanism. The device is configured with several general accounts by factory default, including root/admin, admin/admin and guest/guest, which can be used by customers to initially configure OLT. Customers need to create, delete and modify the login account and password of the device according to their own security policies when using the device. We do not recommend using the factory default username and password in the operation network.

The device retains a debugging account for assisting customers in debugging and solving problems, and this account can also be used by customer to find the forgotten password when they forget the login password of OLT. However, the account no longer uses the general password, and the password is calculated and generated according to the unique identification information of the customer's OLT. Only when the customer provides the information of unique identification code in conjunction with the special password generation tool can the password be generated. The password of each OLT is different, which will better ensure the safety of the device.

And C-Data's quite polite, under the circumstances, correction of the record goes on like that. As a result of the fact that Pierre Kim and Alexandre Torres misinterpreted what they saw and thus chose not to first confront C-Data with their findings, their vulnerability report was full of glaring inaccuracies. The good news is that their mis-disclosure didn't actually put all of those C-Data customers' networks at risk, because they never were at risk. C-Data understood that any default access credentials need to be constrained to the device's local serial configuration port.

But purchasers of counterfeit C-Data equipment were not so fortunate. The screenshot (below) depicts a successful login to the counterfeit C-Data device over the network, not locally, using the panger123 / suma123 username/password credentials, to which the counterfeit device declares "Supperuer successfully!" -- presumably attempting to say "superuser". Meaning full remote administrative access has been granted using secret credentials buried in the firmware of that counterfeit high-end C-Data equipment.

```
$ telnet [ip]
*****
      Command Line Interface for EPON System
      Hardware Ver:  V1.2
      Software Ver:  V1.2.2
      Created Time:  Mar 12 2018 06:54:24
      Copyright (c) 2015-2020 All rights reserved.
*****
Username:panger123
Password:suma123

Entry Supperuer successfully!

epon@
alarm                - setting system alarm
best-sys              - configure sys information
epon-workmode         - configure EPON working-mode
ethernet-ring         - configure rapid ring
igmp-snooping         - configure IGMP Snooping
interface             - interface type
ipconfig              - configure the system IP address
logout                - exit the CLI system
mac-address-table     - ctrl-card dynamic mac address table management
mirror                - configure switch mirror
onu-auth              - configure authentication mode for Olt
ping                  - net ping
port-isolate-group    - create port-isolate-group, you must enable port-isolate-mode for group
rmon                  - configure RMON
rstp                  - rapid spanning tree protocol configuration
```

Where does one purchase such equipment? Clearly not from C-Data or any reputable reseller.

So this bring us to a really interesting issue that we haven't touched upon during the nearly 15 years of this podcast: Counterfeit Networking Equipment.

And as it happens, F-Secure Labs just published a beautiful piece of their recent research titled: "The Fake Cisco: Hunting for backdoors in Counterfeit Cisco devices."

<https://labs.f-secure.com/assets/BlogFiles/2020-07-the-fake-cisco.pdf>  
<https://labs.f-secure.com/publications/the-fake-cisco/>

In the introduction to their report, F-Secure paints the picture of this often overlooked issue. They wrote:

Producing counterfeit products is, and always has been, a great business if you don't mind being on the wrong side of the law. There's no need to invest in a costly R&D process, and no need to select the best performing and looking materials; the only criterion is the cost of manufacture. This is why we see many imitations of expensive products on the market, and are likely to continue to see them being made and sold at a fraction of [the] original's price.

[I suppose Rolex watches have become a cliché counterfeited item and counterfeits of expensive women's handbags are another that we hear about.]

Network hardware designed, manufactured, and sold under the Cisco brand is a perfect example of this. Having an excellent reputation because of their great engineering, these products sell at a premium price point. Naturally, this encourages some to try and produce counterfeits as it's a way of making easy money. Stories of such exploits abound in the media: a gang reportedly exporting US\$ 10 million worth of gear to the US, the FBI seizing shipments of fake hardware, and court rulings being issued to stop the manufacturers. What does Cisco do to combat fraud? Actually, a lot. Cisco has a dedicated Brand Protection organization whose purpose is to defend against counterfeit and gray market activities. They partner with customs teams and regional governments all over the world with success. In April 2019, they seized \$626,880 worth of counterfeit Cisco products in one day. However, despite successful operations, Cisco hasn't been able to stop fraud fully. If there's an opportunity to make a fast buck, there'll always be someone willing to take the risk.

In fall 2019, an IT company found some network switches failing after a software upgrade. The company would find out later that they had inadvertently procured suspected counterfeit Cisco equipment. Counterfeit devices quite often work smoothly for a long time, which makes it hard to detect them. In this particular case, the hardware failure initiated a wider investigation to which the F-Secure Hardware Security team was called and asked to analyze the suspected counterfeit Cisco Catalyst 2960-X series switches. This initiated a research project with the following goals:

- Verify no extra functionality such as "backdoor access" was introduced.
- Understand how and why counterfeit devices bypass the platforms authentication security control.

Naturally, as it's not easy to tell genuine and counterfeit devices apart, to verify whether any kind of "backdoor" functionality existed was also not easy, as it required a considerable amount of technical investigative work. Ultimately, we concluded, with a reasonable level of confidence, that no backdoors had been introduced. Furthermore, we identified the full exploit chain that allowed one of the forged products to function: a previously undocumented vulnerability in a security component which allowed the device's Secure Boot restrictions to be bypassed.

This report is 39 pages and I found it utterly fascinating. Since I think many of our listeners will, too, I have the link to the PDF in the show notes and a made it this episode's GRC.SC shortcut, so <https://grc.sc/776>

<https://labs.f-secure.com/assets/BlogFiles/2020-07-the-fake-cisco.pdf>  
<https://grc.sc/776>





Which one is the fake?

As it happens, though F-Secure closely examined two devices that were Cisco counterfeits, they were both running authentic Cisco firmware and software. But Cisco's firmware and software has powerful counterfeit hardware detection built in which prevents it from running on non-Cisco hardware, and the two devices took very different approaches to circumvent Cisco's elaborate boot-time software authentication.

The first counterfeit contained "add-on" circuitry which exploited a race condition in the boot ROM code to bypass its software verification. It did this by intercepting EEPROM control signals, replacing certain bytes in the image being loaded to modify the software's behavior. It appears the processor's printed circuit board in this unit was an exact copy of Cisco's without modification.

Where the first counterfeit received what amounted to post-manufacturing add-on circuitry, the printed circuit board design of the second counterfeit was changed to incorporate the modification made to the first counterfeit, replacing the EEPROM completely with an unknown integrated circuit. This signified a considerable resource investment in design, manufacture, and testing of such forged products compared to the lower-cost ad-hoc approach used by the first counterfeit. The board layout and silkscreen labelling similarities also suggested that the people behind this forgery might have either had access to Cisco's proprietary engineering documentation, such as printed circuit board design files in order to be able to modify them, or they invested heavily in the complex process of replicating the original board design files using only genuine circuit boards as a guide.

So...

In the case of the C-Data counterfeit, a seriously dangerous remotely accessible backdoor was definitely installed into the counterfeit devices. In the case of the extremely elaborate Cisco counterfeits, all of the ingenuity was expended in creating a virtually indistinguishable clone of the original, and then engineering around Cisco's detection that its prized network operating system was running of counterfeit hardware.

