Security Now! #627 - 09-05-17

Sharknado

This week on Security Now!

(Although there are an unbelievable FIVE Sharknado movies, this will be the first and last time we use that title for a podcast!) This week we have another update on Marcus Hutchins, we discuss the validity of Wikileaks documents, the feasibility of rigorously proving software correctness, nearly half a million people need to get their body's firmware updated, another controversial CIA project exposed by Wikileaks, a careful analysis of the FCC's Title II Net Neutrality public comments comments, a neat two factor auth tracking site, the stupid patent of the month, an example of a vanity top level domain, a bit of errata, where did SpinRite come from?, and ... utterly unconscionable security mistakes made by AT&T in their line of U-Verse routers.

There's an ATM in there!

Security News

Brian Krebs is the PERFECT PERSON to sleuth into Marcus Hutchins' past
https://krebsonsecurity.com/2017/09/who-is-marcus-hutchins/

I was JUST made aware of Brian's work, in his post which I've linked to here. So I have not yet been able to dig into it and absorb it. I will before next week and if there's more, I'll share it. But based upon my quick scan, it appears that Brian has uncovered EVIDENCE and formed his own evidence-based option -- whereas our was just hopeful speculation -- that exactly matches what we have previously suggested might be going on. Specifically Brian writes:

Brian: *"At first, I did not believe the charges against Hutchins would hold up under scrutiny. But as I began to dig deeper into the history tied to dozens of hacker forum pseudonyms, email addresses and domains he apparently used over the past decade, a very different picture began to emerge.*

In this post, I will attempt to describe and illustrate more than three weeks' worth of connecting the dots from what appear to be Hutchins' earliest hacker forum accounts to his real-life identity. The clues suggest that Hutchins began developing and selling malware in his mid-teens — only to later develop a change of heart and earnestly endeavor to leave that part of his life squarely in the rearview mirror."

So, yay, Brian! Thanks for doing that work! It exactly corresponds with our own gut feeling based upon almost no evidence. And... if Marcus was not yet an adult?... does that help is disposition?

"Yes, we can validate the Wikileaks emails"
http://blog.erratasec.com/2016/10/yes-we-can-validate-wikileaks-emails.html#.WarnprKGOUk

One of our listeners responded to my caution about placing possible unwarranted trust in Wikileaks documents due to the threat of including fraudulent documents among the legitimate docs. (Thank you, listener!)

So this was an older posting from about 10 months ago (October 2016) by Errata Security's Robert Graham. (Our old buddy from the BlackIce personal IDS / firewall days.)

However, WikiLeaks documents are not leaked eMails, so Robert's blog posting is ONLY relevant to eMail... due to a technology originally developed as an anti-spam solution which, as a direct consequence, can also be used to cryptographically (as in "as sure as it's possible to be") validate eMail.

Yes, we can validate the Wikileaks emails

Before I go any further let just remind everyone that this is not a political podcast and that I work scrupulously to keep us from falling into the controversial pit of politics. I have no idea what Robert's politics are, if he is even political, and I couldn't care less. Because this is purely

about technology... but, fair warning, it addresses a potentially hot topic. I cannot shy away from it for that reason. But just remember that that's also NOT why it's being discussed.

Robert writes (in October 2016):

Recently, WikiLeaks has released emails from Democrats. Many have repeatedly claimed that some of these emails are fake or have been modified, that there's no way to validate each and every one of them as being true. Actually, there is, using a mechanism called DKIM.

[DomainKeys Identified Mail]

DKIM is a system designed to stop spam. It works by verifying the sender of the email. Moreover, as a side effect, it verifies that the email has not been altered.

Hillary's team uses "hillaryclinton.com", which as DKIM enabled. Thus, we can verify whether some of these emails are true.

Recently, in response to a leaked email suggesting Donna Brazile gave Hillary's team early access to debate questions, she defended herself by suggesting the email had been "doctored" or "falsified". THAT'S NOT TRUE. We can use DKIM to verify it.

You can see the email in question at the WikiLeaks site: https://wikileaks.org/podesta-emails/emailid/5205. The title suggests they have early access to debate questions, and includes one specifically on the death penalty, with the text:

It's not a smoking gun, but at the same time, it both claims they got questions in advance while having a question in advance. Trump gets hung on similar chains of evidence, so it's not something we can easily ignore.

Anyway, this post isn't about the controversy [Thank you, Robert!], but the fact that we can validate the email. When an email server sends a message, it'll include an invisible "header". They aren't especially hidden, most email programs allow you to view them, it's just that they are boring, so hidden by default. The DKIM header in this email looks like:

```
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed;
        d=hillaryclinton.com; s=google;
        h=from:mime-version:references:in-reply-to:date:message-id:subject:to
        :cc;
        bh=EHIyNFKU1g6KhzxpAJQtxaW82g5+cTT3qlzIbUpGoRY=;
        b=JgW85tkuhlDcythkyCrUMjPIAjHbUVPtgyqu+KpUR/kqQjE8+W23zacIh0DtVTqUGD
        mzaviTrNmI8Ds2aUlzEFjxhJHtgKT4zbRiqDZS7fgba8ifMKCyDgApGNfenmQz+81+hN
        2OHb/pLmmop+lIeM8ELXHhhr0m/Sd4c/3BOy8=
```

How do you verify this is true. There are a zillion ways with various "DKIM verifiers". I use the popular Thunderbird email reader (from the Mozilla Firefox team). They have an addon designed specifically to verify DKIM. Normally, email readers don't care, because it's the email server's job to verify DKIM, not the client. So we need a client addon to enable verification.

Downloading the raw email from WikiLeaks and opening in Thunderbird, with the addon, I get the following verification that the email is valid. Specifically, it validates that the HillaryClinton.com sent PRECISELY this content, with this subject, on that date.

This is another fabulous use of DNS. An eMail domain publishes a PUBLIC key which corresponds to the server's secret PRIVATE key.

google._domainkey.hillaryclinton.com: type TXT, class IN
v=DKIM1; k=rsa;
p=MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCJdAYdE2z61YpUMFqFTFJqlFomm7C4Kk97
nzJmR4YZuJ8SUy9CF35UVPQzh3EMLhP+yOqEl29Ax2hA/h7vayr/f/a19x2jrFCwxVry+nACH1FVmI
wV3b5FCNEkNeAIqjbY8K9PeTmpqNhWDbvXeKgFbIDwhWq0HP2PbySkOe4tTQIDAQAB

When an SMTP server originates an eMail which it has received from one of its clients and is forwarding on their behalf, it takes the ENTIRE eMail envelope -- with date and timestamp, subject, body, everything... and appends to it a cryptographic signature -- signed with the server's secret key.

The receiving SMTP server or as Robert noted, an eMail client in receipt of the eMail, queries the apparent sending domain's DNS for a DomainKey TXT record containing the DKIM public key... and uses this to verify that the unchanged document was originally signed by the server's private key.

Now, this evidence appears to be pretty damning. But this is a security and technology podcast so we need to ask the question: "How sure are we?" Cryptographically, we're absolutely as positive as we could be. However... as we have often discussed, TLS certificates and DKIM certificates ABSOLUTELY RELY upon the assumption that the signing server is uncompromised. But the whole point of this Hillary eMail debacle was that their eMail server WAS compromised. And as we know, if in addition to the 30-some-thousand eMails that were reportedly obtained, the server SMTP 's private DKIM key was also obtained... then we're right back where we started... but now with a somewhat harder to substantiate claim of document doctoring.

Proof that HMAC-DRBG has No Back Doors
https://www.schneier.com/blog/archives/2017/08/proof_that_hmac.html
http://www.cs.princeton.edu/~appel/papers/verified-hmac-drbg.pdf
New research: "Verified Correctness and Security of mbedTLS HMAC-DRBG"
by Katherine Q. Ye, Matthew Green, Naphat Sanguansin, Lennart Beringer,
Adam Petcher, and Andrew W. Appel.

Background (from Wikipedia):
The [NIST] publication contains the specification for four allegedly cryptographically secure pseudorandom number generators for use in cryptography:

- Hash_DRBG (based on hash functions),
- HMAC_DRBG (Based on Hash-based message authentication code),
- CTR_DRBG (based on block ciphers in counter mode), [and]
- Dual_EC_DRBG (based on elliptic curve cryptography).

Dual_EC_DRBG was later reported to probably contain a kleptographic backdoor inserted by the National Security Agency, while the other three random number generators are accepted as uncontroversial and secure by multiple cryptographers.

Abstract: We have formalized the functional specification of HMAC-DRBG (NIST 800-90A), and we have proved its cryptographic security -- that its output is pseudorandom -- using a hybrid game-based proof. We have also proved that the mbedTLS implementation (C program) correctly implements this functional specification. [[That's exactly what we want: proof that both the theory and the practice (the implementation) are solid.]] That proof composes with an existing C compiler correctness proof to guarantee, end-to-end, that the [compiler's output] machine language program gives strong pseudorandomness. All proofs (hybrid games, C program verification, compiler, and their composition) are machine-checked in the Coq proof assistant. Our proofs are modular: the hybrid game proof holds on any implementation of HMAC-DRBG that satisfies our functional specification. Therefore, our functional specification can serve as a high-assurance reference.

[This is THE FUTURE. We're not there yet. But computer programs are just math. They are not flowers and trees and star fields. They are absolutely and utterly subject to pure and perfect rigorous analysis and verification. BUT IT'S DIFFICULT! It's unclear how fast this will happen. Whether it'll happen within our lifetimes. And the exponential growth in AI capability might become a factor "Ask Watson if the program is provably correct." This is why, as a writer of code myself I have always chaffed at the lay statement -- which is sadly true -- that "all software contains bugs." We know better. Trivial software can easily be bug-free. More complex software, like the USShuttle's flight guidance and navigation computer can also be bug-free if it's sufficiently important and then if sufficient resources can be brought to bear to make it so.

My point is... it IS the nature of software to rapidly get insanely complex as it grows in size. It quickly gets completely unknowable and out of control. And trivial, readily-proven-correct software isn't of much utility. So... what's destined to happen over time is that the COST of rigorous software correctness proofs will be falling dramatically. We don't and can't rigorously prove the correctness of our everyday software because it would still be impossibly expensive to do so. And, arguably, we still don't know how. But we're going to.

Mathew & Company's research was a perfect example if taking a very small and thoroughly understandable module -- essentially what would be a subroutine of a far larger complete system -- the HMAC-DRBG -- and rigorously analyze it right to the ground.

Someday... I have no idea when... all of our software will be bug-free. We won't hope it's right, we'll KNOW it's right. Because a descendant of Watson told us so!

... and speaking of bugs in software of critical systems ...

FDA Recalls 465K Pacemakers Tied to MedSec Research
https://threatpost.com/fda-recalls-465k-pacemakers-tied-to-medsec-research/127750/

Get a load of this one! (This will make your chest thump!)

The United States Federal Drug Administration is recalling 465,000 pacemakers to which attackers can gain unauthorized access to issue commands, change settings and maliciously disrupt. Affected are four models manufactured by Abbott Laboratories.

According to the FDA, the recalls of affected pacemakers are tied to research by MedSec Holdings that originally brought St. Jude Medical equipment flaws to light about a year ago. Abbott Laboratories acquired St. Jude Medical in January.

[[snip]]

The U.S. Industrial Control System Cyber Emergency Response Team (ICS-CERT) cites three vulnerabilities in its advisory affecting Abbott Laboratories' pacemakers manufactured prior to August, 2017 that include Accent/Anthem, Accent MRI, Assurity/Allure, and Assurity MRI.

The highest rated of the three vulnerabilities (CVE-2017-12712) is related to the pacemaker's authentication algorithm, authentication key and time stamp that can be compromised or bypassed. That could allow a nearby attacker to issue unauthorized commands to the pacemaker via RF communications.

An additional bug (CVE-2017-12714) could significantly reduce the battery life of a pacemaker. "The pacemakers do not restrict or limit the number of correctly formatted 'RF wake-up' commands that can be received, which may allow a nearby attacker to repeatedly send commands to reduce pacemaker battery life," according to the ICS-CERT advisory.

A third flaw (CVE-2017-12716) found in Accent and Anthem pacemakers is related to the fact the devices transmit unencrypted patient information via RF communications to programmers and home monitoring units. Also problematic is that both pacemakers store patient data in clear text on the devices themselves.

According to the recall information: "These vulnerabilities could be exploited via an adjacent network. Exploitability is dependent on an attacker being sufficiently close to the target pacemaker as to allow RF communications."

This is the second time Abbott Laboratories has updated the heart implants. Last October, as a result of a U.S. government probe into potentially life-threatening hacks that could prematurely drain pacemaker batteries, St. Jude recalled a number of implanted heart devices. According to a Reuters report [get this!], premature battery depletion have been linked to two deaths in Europe.

Mitigation will require patients to visit their doctor for a short-range wireless update. Abbott warns the firmware updates should be approached with caution. "Like any software update, firmware updates can cause devices to malfunction," it states. A botched update could result in a loss of settings to complete loss of device functionality.

WikiLeaks: CIA Caught Planting Malicious Software In Windows – Codename 'Angelfire'
http://investmentwatchblog.com/wikileaks-cia-caught-planting-malicious-software-in-windows-codename-angelfire/

A team of hackers at the US Central Intelligence Agency, allegedly used a Windows hacking tool against its targets to gain persistent remote access.

As part of its Vault 7 leaks, WikiLeaks, last Thursday, revealed details of a new implant developed by the CIA named "AngelFire", to target computers running the Windows OS.

The AngelFire framework implants a persistent backdoor on the target Windows computers by modifying their partition boot sector.

AngelFire framework consists five following components:

1. Solartime — modifies the partition boot sector to load and execute the Wolfcreek (kernel code) every time the system boots up.

2. Wolfcreek — a self-loading driver (kernel code that Solartime executes) that loads other drivers and user-mode applications

[So what we have here is an intelligence and law enforcement developed, Windows-specific Rootkit system for enabling remote surveillance.]

3. Keystone — a component that utilizes DLL injection technique to execute the malicious user applications directly into system memory without dropping them into the file system.

4. BadMFS — a covert file system that attempts to install itself in non-partitioned space available on the targeted computer and stores all drivers and implants that Wolfcreek starts.

5. Windows Transitory File system — a new method of installing AngelFire, which allows the CIA operator to create transitory files for specific tasks like adding and removing files to AngelFire, rather than laying independent components on disk.

According to a user manual leaked by WikiLeaks, AngelFire requires administrative privileges on a target computer for successful installation.

The 32-bit version of implant works against Windows XP and Windows 7, while the 64-bit implant can target Server 2008 R2, Windows 7.

[Once again we've seen overblown and clickbait coverage of this in the press. For example I read one report which opened with: "All Windows machines have been infiltrated by the CIA under a project codenamed 'Angelfire' – allowing the U.S. government to load malicious programs onto a persons computer without their knowledge."

No one suggests that this should ever be perpetrated upon US Citizens in contravention of US law and the US constitution. And I'm not intending to defend the CIA. I'm just reminding us that

what we don't know is how this tool was deployed and employed. It may very well be a huge asset for foreign intelligence gathering... and it may never have been deployed illegally without warrant against US citizens.

We don't know... but it sure looks like a slick piece of work, and it's nice to know that "our side" has some skills! :)

98.5% of unique net neutrality comments oppose Ajit Pai's anti-Title II plan
Besides form letters, ISP-funded study finds almost no support for repealing rules.
https://arstechnica.com/tech-policy/2017/08/isp-funded-study-finds-huge-support-for-keeping-current-net-neutrality-rules/

[This falls under the category of: "Be careful what you ask for!"]

ArsTechnica: A study funded by Internet service providers has found something that Internet service providers really won't like.

The overwhelming majority of people who wrote unique comments to the Federal Communications Commission want the FCC to keep its current net neutrality rules and classification of ISPs as common carriers under Title II of the Communications Act, according to the study released today.

http://www.emprata.com/reports/fcc-restoring-internet-freedom-docket/

The study (available here) was conducted by consulting firm Emprata and funded by Broadband for America, whose members include AT&T, CenturyLink, Charter, CTIA-The Wireless Association, Comcast, NCTA–The Internet & Television Association, the Telecommunications Industry Association (TIA), and USTelecom.

When Emprata analyzed all 21.8 million comments, including spam and form letters, 60 percent were against FCC Chairman Ajit Pai's plan to repeal the Title II classification, and 39 percent supported the repeal plan. But the numbers shifted starkly in favor of keeping the Title II rules when excluding duplicates in order to analyze just unique comments written by individuals.

Emprata wrote:

There are considerably more "personalized" comments (appearing only once in the docket) against repeal (1.52 million) versus 23,000 for repeal. Presumably, these comments originated from individuals that took the time to type a personalized comment. Although these comments represent less than 10 percent of the total, this is a notable difference.

That amounts to 98.5 percent of personalized comments supporting the current rules.

Form letters constitute the majority of comments on both sides. This was especially pronounced in the case of anti-Title II comments:

The overwhelming majority of comments for and against repealing Title II are form letters

(pre-generated portions of text) that appear multiple times in the docket. The form letters likely originated from numerous sources organized by groups that were for or against the repeal of Title II. Form letters comprise upwards of 89.8 percent of comments against Title II repeal and upwards of 99.6 percent of the comments for Title II repeal.

Emprata said that it was contracted by Broadband for America "to perform an independent and unbiased analysis of the comment data received by the FCC in response" and that Emprata itself "does not have a vested interest in whether Title II is repealed or not."

Broadband for America provided a link to the study on its homepage. The statement on the group's homepage today did not mention the public's broad opposition to repealing Title II net neutrality rules, saying only that the "report by expert data analytics firm reveals unprecedented volume and clutter in the docket."

Broadband for America's homepage advocates for overturning the rules, saying that "Repealing Title II utility Regulations Will Strengthen the Internet—utility regulations deter investment in networks and put Internet jobs at risk."

ISPs support net neutrality despite their stance against using the FCC's Title II authority to enforce net neutrality rules, the group says.

Two Factor Auth (2FA)
https://twofactorauth.org/
List of websites and whether or not they support 2FA.
Super-cool display breaking out what forms of 2FA they support:
SMS
TOTP, Hardware token, etc.

Stupid Patent of the Month: JP Morgan Patents Interapp Permissions
https://www.eff.org/deeplinks/2017/08/stupid-patent-month-jp-morgan-patents-interapp-permissions

Stupid Patent of the Month: JP Morgan Patents Interapp Permissions:

We have often criticized the Patent Office for issuing broad software patents that cover obvious processes. Instead of promoting innovation in software, the patent system places landmines for developers who wish to use basic and fundamental tools. This month’s stupid patent, which covers user permissions for mobile applications, is a classic example.

On August 29, 2017, the Patent Office issued U.S. Patent No. 9,747,468 (the ’468 patent) to JP Morgan Chase Bank, titled “System and Method for Communication Among Mobile Applications.” The patent covers the simple idea of a user giving a mobile application permission to communicate with another application. This idea was obvious when JP Morgan applied for the patent in June 2013. Even worse, it had already been implemented by numerous mobile applications. The Patent Office handed out a broad software monopoly while ignoring both common sense and the real world.

The full text of Claim 1 of the ’468 patent is as follows:

A method for a first mobile application and a second mobile application on a mobile device to share information, comprising:

the first mobile application executed by a computer processor on a mobile device determining that the second mobile application is present on the mobile device;

receiving, from a user, permission for the first mobile application to access data from the second mobile application;

the first mobile application executed by the computer processor requesting data from the second mobile application; and

the first mobile application receiving the requested data from the second mobile application.

That’s it. The claim simply covers having an app check to see if another app is on the phone, getting the user’s permission to access data from the second app, then accessing that data.

.MONTBLANC
We rarely see entire top-level domains killed, but .MONTBLANC was removed on Friday. Apparently they no longer wanted it.
IANA whois updates @ianawhois
IANA whois: deleted TLD MONTBLANC iana.org/domains/root/d…
https://www.iana.org/domains/root/db/montblanc.html

Errata

- The Wrath of Kahn: On the Kirk / Kahn story it was Lt. Savik (Kirsty Ally) who was quoting regulations to Kirk.
    - Absolutely right... and she was junior and it was Spock who walked her back stating that the captain was well aware of fleet regulations.

- @SGgrc Khan may have misused trust to get Kirk but he didn't change his ship's remote access code. Kirk remoted in and lowered his shields.

Miscellany

- AndrzejL @12nr1981
- @SGgrc Hi Steve, Can you provide me with your "mustacheless" photo? Can't grab it from latest podcast - microphone covers part of your face.

SpinRite

- STEVE HOLDEN @sholden
  .@SGgrc I wanted to let you know that SpinRite was able to help me recover some 2006 podcasts from the Eurythimics chvrchespodcast.com/2016/12/found-…
  http://www.chvrchespodcast.com/2016/12/found-the-eurythmics-podcast-from-2006-1.html

- What IS SpinRite?

Closing The Loop

- Marc Beaupré-Pham @marcbeaupre
  @SGgrc I would assume that you would recommend the Lastpass Authentication over others

- Simon Pickup @SimonPickup
  @SGgrc #SN626 XON/XOFF live on. I use it every day pressing Ctrl-S to pause scrolling tty output on an xterm. Surely you do too?

- What the hell is "Getting your car Smogged?"

SharknAT&To

https://www.nomotion.net/blog/sharknatto/

When I encountered this, and the near-criminal negligence this demonstrates, or should be regarded as, on the part of AT&T, I was STUNNED. And it immediately became our focus for this week. Not only as yet another useful walk through showing just how much can go wrong, but also to advise this podcast's listeners who may be AT&T U-Verse subscribers and might therefore be directly affected, but also to continue and to support the explicit intention of the publication of this research to further highlight AT&T's negligence and the extreme danger that negligence has placed their paying and trusting customers in.

"Sharknado" indeed.

Introduction

When evidence of the problems described in this report were first noticed, it almost seemed hard to believe. However, for those familiar with the technical history of Arris and their careless lingering of hardcoded accounts on their products, this report will sadly come as no surprise. For everyone else, prepare to be horrified.

In all fairness, it is uncertain whether these gaping security holes were introduced by Arris (the

OEM) or if these problems were added after delivery to the ISP (AT&T U-verse). From examining the firmware, it seems apparent that AT&T engineers have the authority and ability to add and customize code running on these devices, which they then provide to the consumer (as they should).

Some of the problems discussed here affect most AT&T U-verse modems regardless of the OEM, while others seem to be OEM specific. So it is not easy to tell who is responsible for this situation. It could be either, or more likely, it could be both. The hope behind writing this is that the problems will be swiftly patched and that going forward, peer reviews and/or vulnerability testing on new releases of production firmware will be implemented prior to pushing it to the gateways. Security through obscurity is not acceptable in today's high threat landscape and this is especially true regarding devices which a) route traffic, sensitive communications and trade secrets for millions of customers in the US, b) are directly reachable from the Internet at large, and c) have wireless capability and therefore have an additional method of spreading infection and releasing data.

Regardless of why, when, or even who introduced these vulnerabilities, it is the responsibility of the ISP to ensure that their network and equipment are providing a safe environment for their end users. This, sadly, is not currently the case. The first vulnerability found was caused pure carelessness, if not intentional all together. Furthermore, it is hard to believe that no one is already exploiting this vulnerability at the detriment of innocents. Which is why this report is not passing Go, not collecting $200, and is going straight to the public domain. The vulnerabilities found here will be ordered roughly from least to most prevalent.

[So there's some trouble here with attribution, but regardless of whether the problems were present in the original ARRIS manufactured device, or were introduced afterward by AT&T, AT&T's end-user customers wind up being the victims.

- 1. SSH exposed to The Internet; superuser account with hardcoded username/password.

It was found that the latest firmware update (9.2.2h0d83) for the NVG589 and NVG599 modems enabled SSH and contained hardcoded credentials which can be used to gain access to the modem's "cshell" client over SSH. The cshell is a limited menu driven shell which is capable of viewing/changing the WiFi SSID/password, modifying the network setup, re-flashing the firmware from a file served by any tftp server on the Internet, and even controlling what appears to be a kernel module whose sole purpose seems to be to inject advertisements into the user's unencrypted web traffic. Although no clear evidence was found suggesting that this module is actually being used currently, it is present, and vulnerable. Aside from the most dangerous items listed above, the cshell application is also capable of many other privileged actions. The username for this access is "remotessh" and the password is 5SaP9I26.

To reiterate the carelessness of this firmware's release, the cshell binary is running as root and so any exploitable command, injection vulnerability or buffer overflow will result in a root shell.

Yes, it is running as root, and trivially susceptible to command injection. Through the use of the menu's ping functionality, and due to not sanitizing parameters, one can execute arbitrary commands through the menu, or escape the menu altogether. An example payload is shown below.

>> ping -c 1 192.168.1.254;echo /bin/nsh >>/etc/shells

>> ping -c 1 192.168.1.254;echo /bin/sh >>/etc/shells

>> ping -c 1 192.168.1.254;sed -i 's/remotessh:\/:\/bin\/cshell/remotessh:\/:\/bin\/nsh/g' /etc/passwd

Now type exit and then reconnect via SSH. The prompt will change from NOS/xxxxxxxxxxxxx to Axis/xxxxxxxxxxxxxxx. At this point the attacker can type "!" and will be given a busybox root shel!.

Please note that the cshell binary was only examined briefly and only until the easiest exploit was found. Judging by the binary's repetitive use of unsafe C functions, one can guess that hundreds of additional vulnerabilities exist. However, we find it highly amusing that the first vulnerability found was so trivial that it looks like it came out of one of those "hacking tutorials" that were popular in the 90's (Google "how to hack filetype:txt").

This is the first and least common vulnerability that was discovered. The number of exposed devices while not as huge as the rest, but it is still quite unacceptable when you realize that these devices directly correlate to people being put at unnecessary risk of theft & fraud.

Censys reports 14,894 hosts which are likely vulnerable. There is no guarantee expressed or implied in terms of this number being all-inclusive however.

- 2&3. Default credentials "caserver" https server NVG599

A HTTPS server of unknown purpose was found running on port 49955 with default credentials. The username "tech" with an empty password field conveyed access to this highly vulnerable web server, which used only a Basic Authorization scheme. The server seems slightly unstable with its authorization capacity, denying access on the first attempt even with valid credentials and eventually completely locking up with an "unauthorized" message. It remains unclear whether this is just poor coding or more security through obscurity, but either is unacceptable.

And there's a trivially exploitable command injection vulnerability here: The exact intended purpose of caserver is unclear, but its implications are not. Caserver is an https server that runs on port 49955 of affected devices (which seems to only be the NVG599 modem). The caserver script takes several commands, including:

- Upload of a firmware image
- Requests to a get_data handler which enumerates any object available in its internal "SDB" databases with a lot of fruitful information
- Requests to a set_data command which allows changes to the SDB configuration

[They show a screenshot of the request which causes command injection, again … as the root user!] Note that for the first request the server will probably reply "You are not authorized to access this page". This can simply be ignored and resubmitting the request shown will yield

command execution. The service can be a little quirky, it locks you out after about 5 requests, a reboot will fix the issue if you are testing and running into this problem. The User-Agent field seems to be required but any string will suffice.

There are countless ways to exploit this, but a few quick and dirty stacked commands using wget to download busybox with netcat (mips-BE) from an http server (no SSL support) and then spawn a reverse shell works well.

Estimating the number of hosts affected was trickier due to the service being on an uncommon port. Host search engines such as Censys and Shodan don't commonly scan for these services or ports. Based on self-collected data, our ballpark figure is around 220,000 devices.

- 4. Information disclosure/hardcoded credentials

The next vulnerability involves a service on port 61001 which will give an attacker a plethora of useful data about the device. The attacker however, will need to know the serial number of the device ahead of time. Once this information is acquired, the request can be made.

Just before the serial number notice the characters "001E46". This number correlates with the model number and is a valid Arris Organizationally unique identifier (OUI). This particular OUI was brute-forced from a list of Arris OUIs obtained at https://www.wireshark.org/tools/oui-lookup.html.

When the correct serial number, OUI, and username/password are submitted as above, the server will hang for several seconds before returning a response. Afterwards, several pieces of invaluable information are returned about the modem's configuration, as well as its logs. The most sensitive pieces of information are probably the WiFi credentials and the MAC addresses of the internal hosts, as they can be used for the next vulnerability.

The hardcoded username/password credentials are bdctest/bdctest. This is the second most prevalent vulnerability but at the moment it is not the biggest threat since the modem's serial number is needed to exploit it. This may change if an attacker were to find a reliable way of obtaining the serial number. If present, an attacker could use the aforementioned "caserver" to retrieve the serial number as well by requesting a valid file present in the webroot other that "/caserver". Once such example of this would be "/functions.lua". Sending a GET request to this file will return the serial number amongst the headers.

This normally would not be advantageous for an attacker since the presence of the caserver service equates to root shell access. However, if the caserver is locked, then this is a method to overcome the lockout since only the path "/caserver" is locked-out.

- 5. Firewall bypass no authentication

[No one is going to believe this! -- Remember: THE MOST PREVALENT OF ALL!]

The most prevalent vulnerability, based solely on the high number of affected devices, is the firewall bypass that is made possible by the service listening on port 49152. This program takes a three byte magic value "\x2a\xce\x01" followed by the six byte mac address and two byte port

of whichever internal host one would like to connect to from anywhere on The Internet! What this basically means is that the only thing protecting an AT&T U-verse internal network device from The Internet is whether or not an attacker knows or is able to brute-force the MAC address of any of its devices! Note however, that the first three bytes (six characters) of a MAC address are very predictable since they correspond to the manufacturer. Given this an attacker could very well start out with this scheme with the unknowns marked as:

"\x2a\xce\x01\xab\x23\xed\x??\x??\x??\x??\x??"

To make matters worse, this TCP proxy service will alert the attacker when they have found a correct MAC address by returning a different error code to signify that either the host didn't respond on the specified port or that an RST was returned. Therefore, the attacker is able to attack the MAC address brute-force and the port brute-force problems separately, greatly decreasing the amount of keyspace which must be covered. The scheme now looks something like this (guessing last three bytes of MAC):

"\x2a\xce\x01\xab\x23\xed\x??\x??\x??\xaa\xaa"

Followed by (Guessing port, same as a TCP port scan):

"\x2a\xce\x01\xab\x23\xed\x38\x41\xa0\x??\x??"

At which point it is now feasible to for a determined hacker to use a brute force attack. Aside from the brute force approach, there are other methods of obtaining the MAC addresses. Such as the previously mentioned vulnerability, or using a wireless device in monitor mode in order to sniff the wireless client's MAC addresses. Basically, if your neighbor knows your public IP address, you are in immediate danger of intrusion.

Going off of the example above, if the device with MAC address ab:23:ed:38:41:a0 has an http server running on port 80 (with the firewall configured to not allow incoming traffic) and an attacker wants to connect and issue a GET request on the webroot. The command will be:

python -c 'print "\x2a\xce\x01\xab\x23\xed\x38\x41\xa0\x00\x50GET / HTTP/1.0\n\n"' | nc publicip 49152

This will open an unauthorized TCP connection between the attacker and the "protected" web server despite the user never authorizing it.

It is believed that the original purpose of this service was to allow AT&T to connect to the AT&T issued DVR devices which reside on the internal LAN. However, it should be painfully obvious by now that there is something terribly wrong with this implementation. Added to the severity is the fact that every single AT&T device observed has had this port (49152) open and has responded to probes in the same way. It is also important to note that the gateway itself cannot be connected to in this manner. For example, an attacker cannot set the MAC address to that of the modem's LAN interface and the port to correspond to the web configuration console. This attempt will fail. This TCP proxy service will only connect attackers to client devices.

In Conclusion

In 2017, when artificial intelligence runs the largest advertising firm on the Internet, when only last year the largest leaks in American history occurred, and where vehicles are self driving, autonomous, Internet connected, and hacked … why do we still find CGI injections, blank default passwords with root privileged services exposed, and what most will likely term “backdoored” credentials?

Developing software is no trivial ask, it is part of this company’s core services, but carelessness of this magnitude should come with some accountability. Below are some workarounds for the vulnerabilities described in this write-up, the time of full disclosure is gone (mostly), but let the time of accountability begin.

Accountability... or is ok to continuously accept free credit monitoring for vendors, governments, and corporations “accidentally” exposing your privacy and in this case, maybe that of your family’s too?

~30~