Personal Information Privacy

Current best practices for both HIPAA and the PCI are encryption of “data in motion” and “data at rest”. Ever wonder about the transition state, when data stops moving and comes to rest? This presentation will deal primarily with that critical phase when data is no longer protected by its “in-motion” encryption and not yet protected by its “at rest” encryption. My talk this afternoon on “homomorphic encryption” will focus in on the “data at rest” encryption “problem”. Did I say “problem”? Yes. There are fundamental problems with “at rest” encryption as well and only now are a set of robust and practical solutions becoming available. But enough of that particular problem. If you are concerned about the protection of data at rest (and you should be!), come back to this room at 12:25

What is “data in motion” protection? Simply, its transportation security. In the physical world, it’s the armor cars that move our physical money and valuables around. In the electronic world, it’s the protected movement of electrical signals that convey information. Back in the pre-World Wide Web days, it often involved special cables and conduits. Today, security of data in motion almost exclusively through encryption. This can be “block encryption”, such as chat or email, where the entire portion of a communications session is encrypted, or “streaming encryption”, such as SSL or VPNs, where the communications are continuously encrypted.

Block encryption is like a “secret message”, passed from spy to spy. Some of the more popular forms of this type of encryption are Phil Zimmerman’s “Pretty Good Privacy” (PGP), S/MIME, and Off The Record (OTR) protocol. For streaming encryption, we have SSL and TLS [since SSL is still a more well-known term than TLS, I will probably refer to SSL during the rest of this talk – however, please remember that I also mean TLS and when I mention a shortcoming in SSL, the same shortcoming is also present in TLS]. VPNs are another, related form of streaming encryption for data in motion. Some VPNs actually use SSL/TLS, while others use such protocols as L2TP and PPTP. What’s interesting about L2TP and PPTP is that both use the word “tunneling” in their names. Tunnels are conduits that channel and protect thing that pass through them. What passes through them could be cars, trains, water, oil, or….data.

The type of data that we are interested in is personally identifiable information (PII). PII are the attributes that fairly uniquely describe - “you.” Your SSN is PII. Hopefully, there is nobody else in this world with your SSN. Your name is PII, but only when it’s associated with other characteristics that are semi-unique to you. There are plenty of Brad Whiteheads out there. However, the number of Brad Whiteheads out there with my birthdate are hopefully small. Add in where I was born and chances are you’ve found a combination that’s unique to me.

Now, the US Federal government and other governmental agencies have come up with detailed charts that define exactly what combination of what characteristics are officially defined as being PII. Let’s not worry about that. For the sake of this talk, let’s just define PII as anything about you that you wouldn’t want publically available. If somebody has your name, SSN, and DOB, they could open up credit card accounts in your name. Or possibly access your bank account. If you have an account on Ashley

Madison, you might not even want people to know your name. Personally, I don't want anybody to know I'm a member of the Taylor Swift Official Fan Club – it ruins my macho image ☺

These days, with all the online services, the vast majority of our PII enters the digital ecosystem through web sites. Even when the official process eventually requires some type of physical presence or proof, such as applying for a passport or a visa, we start the process off on a web page. So, one of the most important cases of the protection or encryption of data in motion is the transfer of our PII through SSL from the web page where it's collected, to the location where it's finally processed and, perhaps, stored.

SSL is a bit of a miracle. It's a relatively simple protocol for establishing a secure tunnel, or pipeline, between your browser and a website. It's a fairly universally understood standard, so all the browsers and websites, for the most part, are interoperable. It's secure, with enough evidence available that it's easier to circumvent SSL than it is to crack it. It operates independent of the web application layer, so as a programmer I don't have to write SSL-specific code. It's now built into almost every browser and web server, and there are libraries for every popular computer language for those cases when you are writing your own browser, web server, or app. Without this ubiquitous means of protecting data in motion, we wouldn't have developed eCommerce nearly as rapidly as we did and without safe eCommerce, with web would have failed.

So, there is no inherent problem with SSL or TLS. However, like any tunnel, SSL has two end points, where things enter and exit the tunnel or pipe. Entry isn't a problem. We fill out PII on a form in our browser and that information enters the SSL pipe established between our browser and the host web site. Your PII isn't totally safe on your browser: there could be key stroke loggers infecting your machine, or a programming error in the browser or the operating system might give malware the ability to see the web form you are filling out, but these attacks on the privacy of your PII involve a malicious program already being on your computer. By the judicious use of antivirus software, virtual machines, sandboxes, and browsers, you can avoid and/or mitigate malware on your machine.

The problem lies with the exit of the SSL tunnel on the web server end of things. All your PII comes tumbling out in human readable form! For a malicious person, this is a gold mine! Or, since we are talking about pipelines, perhaps a better analog might be an oil well. A "Gusher of PII"!

So, we can hope that our unprotected PII is quickly scooped up and stored in a protected, encrypted database or file system. But you know I wouldn't even bring this up if that were the case! ☺

So, where does our PII become "data at rest" and protected by best practices encryption? Let's follow the trail:

First, your PII may not even make it to the intended website! Large corporations and hackers both use what's known as an SSL Proxy. The SSL Proxy puts itself in the middle between your browser and the website to which you are sending your information. If an SSL Proxy is used for malicious purposes, it's called a "man in the middle" attack. If it's used by corporate IT, it's to help ensure that valuable corporate information isn't being sent outside the corporate firewall, and that the corporate network is

only being used for permitted purposes. Either way, an SSL Proxy pretends to be the destination website. As such, terminates your protective SSL pipe and allows a human to read information being sent through the pipe. If the SSL Proxy is being run by a hacker, they now have all your PII. Even if the SSL Proxy is being run by corporate IT and you are permitted to do personal banking (on your lunch hour of course), you may not want the corporate IT staff having access to your banking account password.

Now, when you reach your intended website, why doesn't all your information go directly into a secure database? For a number of different reasons. First, you need something that knows how to talk SSL to be the end point of the pipeline. Usually, that "something" is a load balancer or a web server. They terminate the SSL pipe and then pass your vulnerable PII to the application server and/or database. While almost all commercial databases can have secure connections, very few have the ability to interact directly with the SSL pipe from a browser.

Second, even if you could directly expose your database to the Web, you almost certainly wouldn't want to, for security reasons. Databases are large troves of concentrated data. Hackers like nothing more than to infiltrate a database and download its contents. For the same reason a football quarterback is protected by a front line of big blockers, you don't want to expose your customer, client, or citizen PII directly to the Internet. You want network firewalls, web servers, application servers, application firewalls, and database firewalls standing between the hackers on the Net and your database.

Third, there's a database performance issue to consider. While websites can easily handle hundreds of thousands of browser requests per second, a database usually can maintain only several hundred or even thousand simultaneous connections. Normally, an application or web server will open up a limited number of connections between the server and the database. This pool of open connections will be used by the server to satisfy web requests. Hundreds and thousands of individual web requests will be sent down each of the database connections in the pool. These may be secure encrypted connections, but not the original SSL connection that was protecting your PII.

For a small website, the SSL pipe may end at the web server. Your PII will then be routed and processed through the web server before it goes into the database.

More commonly, the web site receiving your PII will have a set of devices called "SSL accelerators" in front of the web servers. We've all experienced the problem of slow web servers. For an eCommerce web site like Amazon or eBay, time is quite literally money. Slow web pages translate directly to lost revenue. SSL accelerators were developed to remove the burden of decrypting the SSL connection from the web server, allowing it to handle more purchases. The SSL accelerator has special hardware that can directly decrypt SSL connections and then pass the subsequent customer information, including PII, onto the web server or web servers. So, in almost all cases, the SSL pipeline from your browser, protecting your PII, will stop "at the front door", and it may pass through several other network connections and servers before it becomes "data at rest" and hopefully re-encrypted.

We have recently become aware of another threat to your SSL-encrypted PII. Let me start by saying that we like the company Cloudflare! We believe they are doing more good for internet privacy and security than harm and we applaud their SSL Everywhere efforts. Cloudflare is a content distribution network (CDN) service provider. They have thousands of servers located all over the world. If you are a

Cloudflare customer, then each of these servers will pretend to be your server. This way, a European customer's web request doesn't have to travel all the way to the United States and back. A Cloudflare server located in Europe will pretend to be the US web site and will service the request. Like the SSL Proxy, your SSL-protected information will stop at the Cloudflare server and be decrypted. Here's where it gets interesting. In the case of PII being sent to a web site, Cloudflare will forward the information on to the intended website. Now, if the intended web site uses SSL, Cloudflare will re-encrypt your PII and send it through a new SSL pipe to the intended web site. Your PII is still vulnerable, since Cloudflare employees and their contractors have access to PII when it came out of the original SSL pipe. Add to that the fact that these thousands of Cloudflare servers are located in hundreds of foreign, 3rd party "internet service providers (ISP)"'s data centers. Where the ISP personnel may also have access to your PII. But that's not the "scary" part. If the intended web site does not use SSL, Cloudflare will forward your PII on to the intended web site UNENCRYPTED! Your PII may travel around the world, through multiple networks in plain text format! And there is no way for the sender to know this is happening. The browser will still show a padlock or an Extended Validation Certificate green bar, because the connection to Cloudflare is SSL-encrypted. The customer, citizen, or patient has no way of knowing if Cloudflare is sending their PII through a second SSL tunnel or just as plain text. Let me be clear about this – Cloudflare does not endorse using SSL for only part of the connection. Their instructions specifically warn a web site administrator not to do this if private information being sent. However, we have read numerous instructional posts on common web site administration discussion groups, explaining the "easy way to set up SSL".

So, if your PII spends so much of its time in human readable form on large websites, why aren't data breaches even more common than they are? Well up until recently, web sites were hosted in corporate data centers and controlled by corporate IT security teams. Your online banking information was being processed by a web server running on a computer in the bank's corporate data center. Corporate IT security teams made sure than only properly authorized bank employees could access the web servers. There were audit trails that logged how each employee accessed corporate resources. The IT security teams watched the whole data center and they, in turn, were watched by other auditors, again monitoring the whole data center and network. All requests coming to and from the Internet could be watched and unusual activities detected. There is always the possibility of insider threat, but with the access controls and audit trails, it was difficult for anybody, IT or banker, to access information surreptitiously. We know corporate data centers aren't invulnerable; we need only look to the J.P. Morgan breach last November where 76 million customer PII records were lost. Or Sony Pictures Entertainment in 2014, where 12 terabytes of employee personnel records, financial statements, actor's salaries, and executive personal emails were publically disclosed by hackers.

Now, let's imagine that all your financial and health PII were hosted in a foreign data center in China or Russia. Feeling a bit nervous? All your unencrypted, human readable PII is being switched between different computers in a hostile, foreign data center.

That, in a nutshell, is “The Cloud”! Now, let me take a moment to define what I mean when I refer to the cloud. There are about as many different definitions of The Cloud as there are commercial IT companies. When I say The Cloud, I’m talking about the companies that provide data processing capacity and data processing services for other companies. This form of The Cloud is often referred to as “Infrastructure as a Service (IaaS)”, or “Platform as a Service (PaaS)”. Examples of this type of Cloud are Amazon Web Services, Google Cloud Platform, Microsoft Azure, and RackSpace, just to name a very few of the largest and best known names. Now, don’t get me wrong. Cloud services are fantastic and are directly responsible for the continued growth of internet technology and commerce. I have been the CIO for a large corporation, and an enterprise architect for several large government agencies, both here in the US and abroad. Each year, you have to project what new services you are going to deliver and what computing power each service will require. You then have to convince the other members of the C suite or the government department to fund your estimates. If you are wrong in your estimates or your budget isn’t fully funded, you may not have the capacity to run the company or agency’s IT services successfully. Or, let’s say there’s a seasonal spike in your computing needs. Why buy sufficient servers and network equipment to handle the spikes and then let it lie idle during the off-season. Then there is always what we call the “Slash Dot” effect. What happens if your web site becomes insanely popular overnight? Do you try to rush out and buy new servers? And then what do you do with them in two weeks when your popularity falls?

The Cloud is a reasonable answer to all these situations. As a CIO, you can be conservative in your budgeting, knowing that you can always buy additional capacity from a cloud vendor. For the CIO of a seasonal company or agency, you can build out your own data center to handle the steady-state requirements, and then use the cloud for your peak needs. Amazon and Google can bring new server instances online in a matter of minutes, allowing you to respond to your new overnight popularity in…well….overnight! So, the cloud and third party data centers are good. BUT…and it’s a big caveat, cloud users have to engineer their websites and services specifically for the Cloud environment.

A cloud data center has to be viewed as a hostile environment. One in which you have almost no visibility or control. In addition to your web site, there are thousands of other customers and web sites running in the same data center. And unlike a colocation data center, you are intimately sharing hardware and networks with these other customers. Your web site, collecting your customer or citizen’s private information may be running on the very same server as several other web sites.

Let’s consider where you are vulnerable in a Cloud data center. First, a cloud data center is like a watering hole in the jungle. Lions, tigers, and other predators have long staked out watering holes as “target rich environments”. Since the IP address ranges of the cloud data centers are well known, hackers are constantly “rattling the front door” as it were, looking for vulnerabilities. Still, in a corporate data center you have the same threat. Data centers are targets; cloud data centers are just larger targets.

In a number of large cloud providers, your customer's SSL pipe ends at the SSL Accelerators at the front of the data center. From there on, the privacy of your customer's data is protected by "virtual local networks (VLANS)". VLANs, as implemented by most network vendors such as Cisco and Juniper, are pretty good protection. However, keep in mind two things; first, your customer's PII goes through a transition from SSL to VLAN at this point and is therefore vulnerable to human collection and review. Second, while we tend to trust the big network equipment companies, a backdoor in Juniper's network code was recently uncovered by the FBI. Huawei is a large Chinese vendor of networking equipment. The US Department of Defense is prohibited from buying Huawei networking equipment because of observed irregularities in their products. Huawei equipment tends to be less expensive than US domestic vendors. Do you trust your cloud providers to have the same economic discipline as the DoD?

As I said before, your web site may very well be running on the same physical server as other users, perhaps even hackers. Like the VLAN software, virtual machine software CAN be fairly secure, if configured correctly. The SANS Institute lists VM misconfiguration as the greatest risk in the use of VM technology in the cloud. I believe the larger cloud providers have locked down configurations, to avoid these types of issues. You will want to confirm the same level of details with smaller cloud vendors. One form of "attacks" that VM technology really can't defend against are "side channel" attacks. In a virtual machine environment, several virtual computers are sharing the memory and CPU resources of a physical computer. As I said, the VM software keeps each VM isolated and unable to directly observe any other virtual machine. However, the possibility of "indirect" observation exists. Through timing checks, one VM can detect when a neighbor VM makes extensive use of the physical server's CPUs. While I don't believe this is a serious threat, I'm always amazed at what the hackers can glean from these types of observations. In 2012, a group of researchers used a VM timing attack to copy data being decrypted on a co-resident VM running under the Xen virtual machine hypervisor, the same VM software used by Amazon for their cloud services, and using the standard libgcrypt library. Xen and libgcrypt have been updated to defeat this particular attack, but then that was 4 years ago.

Truthfully, the biggest threat to your PII in the cloud is the insider threat. We trust the Google, Microsoft, and RackSpace system administrators because their employers have told us they are trustworthy. Look how well this worked for the FBI, the NSA and the CIA. These are intelligence community organizations that routinely subject their employees to surveillance and polygraph testing. Yet, both Aldrin Ames (CIA) and Robert Hanssen (FBI), sold information to the Russian intelligence service. Chelsea Manning, US Army Intelligence, collected and turned over 250,000 classified and sensitive documents to Wiki Leaks. Edward Snowden, an NSA contractor, stole perhaps as many as 1.7 million top secret and sensitive documents from the US DOD, the British GCHQ, and the Australian Intelligence service.

This person is a contractor, working for your cloud provider. Just kidding!!! Still, as we've seen, system administrators have access to your plain text PII at the SSL accelerator – each SSL accelerator has what's called a "spanning port". It's intended for diagnostics and chaining accelerators together. All the decrypted traffic being handled by the accelerator streams past the spanning port. Likewise, each network switch that carries your data has a similar spanning port. Finally, the system admins have access to all the data and software running on your virtual servers. Again, please don't get me wrong. The vast majority of system administrators are dedicated, trustworthy individuals. Still, Carnegie Mellon

University's Software Engineering Institute has an Insider Threat team that, as of 2014, had a database of over documented 1000 insider thief incidents, at both corporate and cloud data centers. It only takes one bad apple to spoil your company's financial outlook.

So, to summary the threat: Despite the use of the best practice of encrypting data in motion with SSL, private PII will be present and processed on your web site in plain text. It is vulnerable to theft and manipulation by other cloud customers, and anonymous cloud provider employees and contractors.

What can you do to mitigate this vulnerability? There's a simple solution – client-side encryption. With client-side encryption, you encrypt the data in the browser, before sending it through SSL, to the web site. When the PII reaches the end of the SSL pipe at the SSL Accelerator or the web server, it will still be encrypted. You, the web site developer, will be able to decrypt and process only the information you need, and only when required. Encrypted PII can be directly inserted into a database without exposing it to hackers or rogue administrators.

At this point, you will truly be protecting your customer or citizen's PII to the best of your ability. You will also be significantly reducing the risk of data loss and breaches. With many US states requiring paid credit protection for victims of data breaches, you may very well be saving your company from significant losses or potentially future bankruptcy.

So why hasn't client-side encryption a standard practice today? As you might imagine, a lot of it's for historical reasons. Back when Netscape developed SSL, there were no cloud providers. Web sites were in private data centers, owned and operated by people trusted with the type of information being sent through SSL. Nobody saw a need for more than SSL. So, we have ubiquitous support for SSL in all our browsers, but no client-side data encryption support. If fact, until very recently, browsers did not even give web developers the tools needed to do client-side encryption. Encryption is highly dependent on random numbers and until the Web Crypto standard recently was adopted by the major browsers, JavaScript developers didn't have a source of good random numbers inside the browser. We view the adoption of the Web Crypto standard as a sign that the shortcomings of SSL encryption are being recognized and addressed. As a developer and user of client-side encryption, my company couldn't be happier with this trend.

Even if browsers had provided the building blocks such as random numbers, up until recently browsers didn't have the "horsepower" to do client-side encryption. Fortunately, both Google and Firefox have gotten into a "horsepower" race and browsers are being much faster. Couple that with Elliptical Curve Cryptography and Diffie-Hellman Key Exchange, instead of the older and much slower RSA encryption, and developing client-side encryption has become feasible.

In the past, if you couldn't write web pages with client-side encryption, you could always write a browser plug-in. The problem there of course was that unless you were Adobe, you couldn't get Microsoft or Firefox to package your plug-in with their browser. Instead, customers or patients had to download and install your browser plug-in from your web site. Not something my mother is capable of doing. Also, as the web site administrator, think about what a headache a custom plug-in is. You have to maintain separate versions for each browser family and you have to constantly test it against each browser release. Add in the need for adjustments for the operating system (Windows, Mac, Linux), and you can see why only a large company like Adobe would want to do plug-ins. Finally, most of the smartphone and tablet browsers don't accept plug-ins.

Even if you did write your own client-side encryption library, let's face it, encryption is hard to do right. Do you sign and encrypt or encrypt and sign? What are your padding characters? Should nonces be monotonically increasing series or random? Which algorithms should you use? Even hackers don't get this right. There was one ransomware program, TeslaCrypt, which used symmetric encryption and then left behind the "public key" – which of course is the only key and can decrypt all the files being held for ransom. [Note – As of March 2016, it appears that a new version of TeslaCrypt is being used, with the encryption flaws of the original version corrected – please use caution when browsing the Internet and opening emails. Use a sandbox or virtual machine, and make offsite, multiple copy backups!]

But these were the bad old days. Today, browser vendors are incorporating the tools and functions required for client-side encryption. Several excellent open source and commercial libraries are available that "do encryption right" and relieve the web developer from having to guess how to properly implement security. And just in time. Like SSL was responsible for successful and safe eCommerce on the web, I think that client-side encryption will be the key to safe and successful Cloud operations.

Now are this point, somebody generally asks "Are you saying we should stop using SSL and switch to client-side encryption?" The answer is a resounding No, we need both. First, SSL actually three separate functions. As we've discussed, it provides encryption protection. It also provides authentication. We know we are talking to the intended web site because of its SSL certificate. That authentication can also go two ways. In the same way the server proves its identity by sending our browser a copy of its certificate, it can request a certificate from our browser to confirm our identity. Finally, SSL can provide trust. To obtain an extended validation certificate, an "EV" certificate, a company or individual has to provide identity documents that are validated through independent third party research. So at least you can find a name and contact information for the certificate holder.

As ACM and IEEE members we develop, review, and promote standards in the computing world. As entrepreneurs, we own web sites and use the Internet. It's important that we recognize what's required to safely use the Internet. SSL has served us faithfully for 21 years, but it is no longer sufficient. We now need to supplement it with client-side encryption. I think we have seen here today that HIPAA and PCI best practices are not sufficient if we truly care about our customers, our citizens, our patients, and our own company's financial future.