Methods for Securing Data Transmission

Daniel Truong

June 25, 2017

Abstract

Transmitting secure data over a TCP/IP setting can be computationally laborious, not to mention open to scrutiny and liability by governing bodies such as PCI-DSS and HIPAA. The idea of this paper is to explore the contemporary methods and technologies used towards the secure peer-to-peer transmission of data. From the standpoint of a private practice firm (utilizing a website for medical records and eCommerce purposes), we'll look at the following methods (and why they're necessary): using SSL/TLS certificates to secure client sessions (HTTPS), minimizing the storage of sensitive data (I.E. credit card numbers, bank account info), using nonces to safely send out identifiable information, utilizing Git to track code changes & enforce integrity, and using secure file transferring (SCP, SFTP, FTPS) to ensure secure file uploads.

Intro

Everyday, data is digitally transferred amongst people and companies at a very fast rate. Financial data, medical data, schoolwork, automotive administration (registration, inspections), entertainment; what used to take days and be at the mercy of paperwork and physical mail delivery now takes fractions of a second and is at the mercy of physics and current technology. However, with this versatile method of digital delivery comes greater security threats. What used to be easy to detect and thwart (people stealing mail, pick-pocketing, door-to-door scams) has now evolved into something that is difficult to detect and is faced by millions everyday on the internet: cyberthreats. Hackers and malicious actors have taken to using digital methods to be able to intercept sensitive data and use them for nefarious purposes. Back in 2014, 17.6 million people reported at least one incident of identity theft[1]. Yet, a net total of $298.6 Billion[2] worth of electronic retail sales were reported for 2014. Governments have gotten involved to make certain that their citizens are kept digitally as safe as possible while supporting the robust commerce that internet trade brings along. For the rest of this paper, methods for securing data transmissions will be evaluated. First, standards for which to establish a baseline for what is considered "secure" will be evaluated.

Regulatory Bodies

Due to the amount of attacks that occur on user data every year, regulatory bodies were formed with the purpose of enforcing secure best practices. Financial and medical institutions, as well as eCommerce firms, are usually never exempt from the policies set forth by these regulatory bodies. Historically, violations have resulted in monetary fines for punishment (with criminal penalties reserved for cases that are egregious or performed with malicious mens rea or "state of mind").

PCI-DSS

The Payment Card Industry Data Security Standard (PCI-DSS or "PCI Compliance") is the standard that dictates what steps entities should take to ensure that payment card data (credit and debit cards) is kept safe for all transactions. This standard mostly applies to eCommerce firms who engage in electronic transactions

[1] "17.6 Million US Residents Experienced Identity Theft in 2014." *Bureau of Justice Statistics.* September 27, 2015. https://www.bjs.gov/content/pub/press/vit14pr.cfm.

[2] "E-Stats 2014: Measuring the Electronic Economy." *United States Census Bureau.* June 7, 2016. https://www.census.gov/content/dam/Census/library/publications/2016/econ/e14-estats.pdf.

on a daily basis[3]. Compliance is mandated by the major card brands (Visa, MasterCard, Discover, etc.) and administered by the PCI Security Standards Council[4]. Below are the steps that a firm must take to ensure PCI compliance[5]:

1. Configure a firewall for protecting card data
2. No default passwords (from the vendor)
3. If cardholder data is stored, ensure adequate protection
4. Have payment card data encrypted over an open/public network
5. Utilize anti-virus software with current definitions
6. Inspect systems and applications routinely for vulnerabilities and updates
7. Payment card information is need-to-know information only
8. Codify each client with a unique ID/Key
9. Payment card data is prohibited from physical access
10. Activity pertaining to network and payment card access is accounted for
11. Security systems are routinely tested
12. Establish a security policy and make sure personnel are aware of it

In the U.S., While PCI compliance is not mandated by federal law[6], some states have deferred to the PCI-DSS standards for mandating payment card security. Companies in Nevada (State Law NRS-603A) and Washington (RCW 19.255.020) could be held liable in the event a security breach were to happen. Otherwise, companies would normally hire an outside firm to audit their setup (for smaller companies, they have the option to self-assess).

HIPAA

The Health Insurance Portability and Accountability Act (HIPAA, Public Law 104-191) was passed by the U.S. Congress back in 1996. The goal of HIPAA was two parts: to protect health insurance if an individual loses or changes jobs (Title I) and to dictate the standards needed to make health information private and secure (Title II). In the context of cybersecurity, Title II is the focus that most medical providers follow regarding health information.

There are many different facets that must be adhered to if an entity were to be HIPAA compliant. They involve Technical, Physical, and Administrative safeguards[7]. Technical safeguards deal with controls related to Access, Audits and Authentication, in addition to Integrity and transmission security. For instance, if an entity installed a new computer for a medical worker, it must be: accessed with a unique username & password, have auto-logoff capabilities, have it's network traffic encrypted and be audited frequently (among other technical safeguards).

For physical safeguards, they include workstation security and usage, device/media controls and access controls to the facility housing the usable equipment. This facet of responsibility includes making sure old computing equipment are disposed of properly, door locks on the facility and policies governing computer usage (but not limited to those actions). Finally, administrative safeguards deal mostly with usage policy and governance. This includes training, designated responsibility of security-related personnel, incident management and business associate management (what external entity will have access to the health care data?).

Enforcement of HIPAA is done through the U.S. Department of Health and Human Services, via the Office

[3] "Executive Summary." *PCI DSS Compliance.* Accessed June 30, 2017. http://pcidsscompliance.net/.

[4] "About Us." *PCI Security Standards Council.* Accessed June 30, 2017. https://www.pcisecuritystandards.org/about_us/.

[5] "PCI DSS Quick Reference Guide." *PCI Security Standards Council.* October 2010. https://www.pcisecuritystandards.org/documents/PCI%20SSC%20Quick%20Reference%20Guide.pdf.

[6] Ellis, David. "PCI DSS Compliance FAQ." *Security Metrics.* May 16, 2014. http://blog.securitymetrics.com/2014/05/pci-faq.html.

[7] "Summary of the HIPAA Security Rule." *HHS.gov Health Information Privacy.* July 26, 2013. Accessed June 30, 2017. https://www.hhs.gov/hipaa/for-professionals/security/laws-regulations/index.html.

for Civil Rights (OCR[8]). Between 2003 and 2017, over 150,000 cases of HIPAA violations were reported[9]. A majority of them faced fines for punishment[10]. A small subset of cases were sent to the US Department of Justice for criminal violations of HIPAA. Civil punishment ranges from $100 per violation (if done unknowingly) to $50,000 per violation (if done knowingly and wasn't corrected beforehand). Criminal penalties range from fines between $50,000 and $250,000 and imprisonment between 1 to 10 years. The most severe punishments are normally reserved for cases where the perpetrator showed malicious intent to utilize the data (personal gain, harm).

GLBA

The Gramm-Leach-Bliley Act (GLBA, US Code 6801) was passed by the U.S. Congress back in 1999. GLBA is mostly concerned with safeguarding the privacy of personal financial data (such as social security numbers, credit scores, etc.). The Financial Privacy Rule of GLBA states that financial institutions must have a privacy policy that informs clients of what information is expected to be collected from them[11]. As a result, institutions are required to have assessments of their security levels (and possible risks attached to them). As a result, GLBA comes into play whenever financial institutions implement online programs that involve client data.

Example Scenario

To illustrate the technologies used for securing data transmissions, we'll use the following fictional scenario. It will involve handling data related to eCommerce, as well as sensitive medical records:

A private practice firm (the firm) would like to encourage their patients to take a more proactive stance on their personal health. To do so, the firm would like the implement an Electronic Health Records system (EHR) to allow their clients to view their medical history online. Furthermore, to improve on customer service to clients, they would like to implement an online billing system that would allow for patients to pay any balance or copays online via credit or debit card.

To implement the system, the firm sets up an onsite LAMP server (Linux - Apache - MySQL - PHP). The firm only wants to serve web content to the outside world. Every other tasks (Server file uploading, Secure Shell access, etc.) will be handled within the firm's internal VLAN. Any features of this EHR will have to abide by rules set forth by PCI-DSS and HIPAA.

HTTPS

To ensure that data is secure over open/public networks, the firm will want to implement HyperText Transfer Protocol Secure (HTTPS, RFC 2818) when serving web content. By default, HTTP serves content over TCP port 80, while HTTPS communicates over TCP port 443. The server admin will want to redirect all port 80 requests to port 443. As a result, when a client goes to access "http://www.thefirm.com" in their web browser, they will instead be redirected to "https://www.thefirm.com".

HTTPS started out using SSL (Secure Sockets Layer) for it's encryption layer when it was defined back in 1994. Due to several security issues that were found with the SSL model, a push was made to secure HTTP traffic using TLS (Transport Layer Security). TLS was developed as the successor to SSL starting at version 3.0. TLS added SHA-256 encryption hashing and protection against cipher-block chaining (RFC 4346).

[8] "Summary of the HIPAA Privacy Rule." *US Department of Health & Human Services: Office for Civil Rights.* Accessed June 30, 2017. https://www.hhs.gov/sites/default/files/ocr/privacy/hipaa/understanding/summary/privacysummary.pdf.

[9] "Numbers at a Glance." *HHS.gov Health Information Privacy.* June 09, 2017. Accessed June 30, 2017. https://www.hhs.gov/hipaa/for-professionals/compliance-enforcement/data/numbers-glance/index.html.

[10] "HIPAA Violations & Enforcement." *American Medical Association.* Accessed June 30, 2017. https://www.ama-assn.org/practice-management/hipaa-violations-enforcement.

[11] Pfleeger, Charles P., Shari L. Pfleeger, and Jonathan Margulies. "Computer Crime." *Security in Computing, Fifth Edition*, 733-744. Pearson Education.

To setup HTTPS on a web server, the firm's server admin will need to get ahold of an identity certificate[12]. The identity certificate, also known as a public key certificate, is used to prove that the firm has ownership of a public key (which is used by entities to decrypt messages sent privately amongst each other). A certificate authority (CA) signs the identity certificate, which gives a layer of integrity in proving that the firm owns the identity certificate. The certificate is then configured for the device that will be used; in this case, the firm's web server. If a client accesses the firm's web site with their web browser, the browser will tell them that the site is secured; in addition, clients can view information about the identity certificate that the site is using to protect with (and make a personal judgement if the site is to be trusted or not).

Secure Third-Party Portal

From the outset, like any other small business starting out, the firm may want to contract their eCommerce payment processing to a third-party such as PayPal, Square or Stripe. There are many advantages towards going this route. Each payment processor has an API that developers (associated with the firm) can implement for their eCommerce needs. Depending on the way the API is used, the firm will not need to store payment card information on their servers (alleviating one of the objectives of being PCI compliant). Plus, payment processors are under all the scrutiny necessary towards ensuring secure payment processing, reducing some of the security overhead that the firm will face.

In theory, how the firm can implement this is as follows: send (or make an HTTP POST request) information only relating to the business service and minimal identification info (Name, address, ZIP code) to the payment processor and redirect client's web browser to the address header associated with the processor. The user then inputs sensitive payment information (such as credit/debit card number, expiration date, billing address) on the processor's website itself. Once the payment processor verifies successful payment, the processor redirects the client's web browser back to the firm's website[13]; concurrently, the processor responds back with minimal information (transaction number, TRUE/FALSE if payment went through) to the firm to prove that the transaction is successful. No sensitive payment card information is stored on the firm's servers at any point.

With this method, the onus of ensuring PCI compliance is mostly on the payment processor. However, the firm still shares some of the onus of compliance as well. It's on the firm to ensure that the payment processor is PCI compliant and to take corrective action if it isn't (such as switching providers if it comes to it). The firm must also know if payment card information is being stored on their servers at any given time and adhere to PCI-DSS guidelines if so.

Nonces

Should the firm go with a third-party payment processor, they will want to utilize nonces to obfuscate patient data with. When it comes to generating a payment transaction, there has to be some information that is used to identify the transaction with the medical services provided (for billing and accounting purposes). However, sending out information such as "STD Medication" or "Cancer Screening Evaluation" and tying them to the patient directly is a blatent HIPAA violation.

Nonces are one-time use numbers that are arbitrary and utilized for denoting specific occasions[14]. In the case of tying medical services to a patient (and setting up a scheme for billing reimbursement), nonces can be used to program the services performed to an arbitrary number (nonce). The nonce can then be sent to the payment processor to codify payment with the services. Outside of the firm, it would be difficult to discern what the nonce actually means. Regulatory firms (ensuring against reimbursement fraud) would need to query the firm directly to prove that services performed are reimbursed correctly.

[12] Durumeric, Zakir, James Kasten, Michael Bailey, and J. A. Halderman. "Analysis of the HTTPS Certificate Ecosystem." *Department of Electrical Engineering and Computer Science.* University of Michigan. 2013.

[13] "Square Checkout Overview." *Square Connect API Documentation.* Accessed June 30, 2017. https://docs.connect.squareup.com/articles/square-checkout-overview.

[14] Pfleeger, Charles P., Shari L. Pfleeger, and Jonathan Margulies. "Cryptography." *Security in Computing, Fifth Edition*, 86-127. Pearson Education.

Federated Identity Management

To make it easy on clients to login to the EHR, the firm may want to contract out to health insurance providers to handle identity management. Health insurance companies are generally under the same amount of scrutiny as health care providers when it comes to HIPAA compliance. This line of thinking is presumable however; not every (or even a majority) of health insurance companies may have an API or even aloow for third-parties to federate access from their user databases. For the sake of optimism, we can assume to be working with an insurance company that does have such a feature. From a patient/client standpoint, that means that they would only need to remember one set of login credentials: to access their health and insurance information.

One way this could be implemented is with SAML (Security Assertion Markup Language). SAML is an XML-based format for exchanging authentication data between parties[15]. This is how user access can be facilitated:

- User goes to the firm's website and requests to login.
- The firm displays an option to login via a predefined list of authorized insurance companies.
- The user selects their insurance company. The user's browser redirects to the insurance company login page.
- The user puts in their username and password. The insurance company does the leg work of confirming correct user credentials.
- Upon successful authentication, the insurance company redirects the user's browser to the firm's website. Simultaneously, the insurance company sends over a SAML assertion (form) that has information about the user (Name, Member ID, Access Privileges, etc.).
- The firm reads information off of the SAML assertion (from the insurance company) and displays information and data according to the user in the SAML assertion.

The problem with going with SAML is the amount of data that gets sent over a SAML assertion[16]. While it may not necessarily contain passwords, it can contain other types of identifiable data related to a user. One would need to ensure that assertions are protected with a security certificate or another type of countermeasure to protect against Man-in-the-Middle attacks. Especially with HIPAA regulations in play, the firm would not want information (from a SAML assertion) to be seen in the open at any time.

Alternatively, the firm could use OAuth for their identification scheme. OAuth (Open Authentication, RFC 6749) is an authentication standard that is used (ubiquitously by big-name social media organizations) for handling user authorization[ˆOAUTH1]. It works differently from SAML in that the firm would be requesting authorization from the insurance company, as opposed to identifiable information via assertion forms. The user would try to login to the firm's site and the firm would redirect the user to the insurance company's login page. From there, the user is then asked to give consent to the firm to access their insurance info for logging in with. Once granted, the firm then talks directly to the insurance company and exchanges the information with them to allow the user to access their services after. The only information passed between the firm and the insurance companies are token strings that are useful for one time use (see the nonce section above).

Again, everything that was mentioned about Federated Identity Management is wishful thinking. In reality, not all insurance companies would have an external API or allow third-parties to access their data in such a manner. Also, patients may not have insurance and decide to use cash or personal payment to reimburse health care. In those cases, the firm will need to setup an authentication server or database to handle individual user logins.

[15] "SAML: The Secret to Centralized Identity Management." *InformationWeek.* November 23, 2004. Accessed June 30, 2017. http://www.informationweek.com/software/information-management/saml-the-secret-to-centralized-identity-management/d/d-id/1028656.

[16] Pfleeger, Charles P., Shari L. Pfleeger, and Jonathan Margulies. "Cloud Identity Management." *Security in Computing, Fifth Edition*, 568-579. Pearson Education.

Secure and Accountable File Transferring

While not directly related to the act of transferring data between the firm and the outside world, it would still be important to keep a lid on the amount of files being changed on the web server. By default, TCP ports 21, 22 and 990 are used for services related to FTP (be it plain FTP, FTPS, SFTP or SCP). Sysadmins should take care to allow access to the aforementioned ports within a specific internal VLAN, never the outside world.

FTP (File Transfer Protocol, RFC 697) is one of the oldest and more ubiquitous ways to transfer files remotely between clients and servers[ˆFTP2]. It operates over port 21. However, FTP does not come with any security. Files transferred over FTP are susceptible to interception via Man-in-the-Middle attacks. Even within a segmented VLAN, it is not recommended to use unsecured FTP to transfer files with[17].

An alternative that implements security could be SFTP (Secure FTP) or SCP (Secure copy). Both methods operate over port 22. How they work is that they allow for file transfers via SSH (Secure Shell) and uses the same authentication scheme as SSH. SSH was designed to provide a secure channel over an open network between the client and the server being accessed. It uses public key cryptography to authenticate remote servers and users (optionally). The difference between SFTP and SCP though is that SCP strictly allows file transfers, while SFTP expands on SCP by allowing permissions manipulation and file removal. SFTP could be thought of as a somewhat interactive implementation of SSH, but designed with directory and file manipulation in mind [18].

FTPS (FTP Secure, RFC 4217) is an extension to the baseline FTP protocol. FTPS adds support for session encryption via SSL or TLS. It is not the same as SFTP. There are two implementations of FTPS: implicit and explicit. Implicit FTPS requires that an SSL/TLS session is started beforehand and attempts to perform normal FTP actions after. The problem with this method is that the recipient server must also have support for FTPS. If the server doesn't have the right setup, the connection is dropped. With Explicit FTPS, the client requests from the server a security method for the session. The server comes up with a security method that is agreed upon by both the client and the server. The connection can then proceed as usual or cut by the user. Implicit FTPS operates off of port 990 while Explicit starts off on normal FTP port 21, then establishes security after.

No matter which file transferring scheme the firm goes with, they will want to implement a system to ensure that files being modified and uploaded are not modified on their journey between the client and server. Enter Git: a version control system for managing changes in files and documents. Git allows it so that the firm could track changes in the codebase of their website[19]. Git could be used to enforce integrity that the files (being changed) are changed as directed (and that a malicious actor hasn't uploaded additional code). With Git, every time a developer makes a change to a file, they must make a note on what was changed before uploading (or committing) the file to their production server. Each change comes with a hash value that corresponds to what was changed. Any deviation in the hash could indicate malicious action. Git also comes with the ability to roll back a change, should malicious code be committed to the firm's codebase.

Conclusion

The last 20 years saw the rise of many different standards and encryption technologies that are used today. As time went on, newer attacks made some of these encryption methods obsolete. These attacks will still continue, mostly due in part to the ease of opportunity and methods for which to carry them out with. The eCommerce machine will continue to invite attacks as long as the motive (monetary gain) is fruitful. But with that comes active awareness of attacks and new methods and technology that could be used to subvert those attacks. At the end of the day, firms (that engage in the transmission of sensitive data) must always be

[17] "Securing FTP using SSH." *Nurdletech.* Accessed June 30, 2017. http://www.nurdletech.com/linux-notes/ftp/ssh.html.

[18] Barrett, Daniel J., and Richard E. Silverman. *SSH, the secure shell: the definitive guide.* Cambridge: OReilly, 2001.

[19] "Tech Talk: Linus Torvalds on git (Hosted by Google)." *YouTube.* May 14, 2007. Accessed June 30, 2017. https://www.youtube.com/watch?v=4XpnKHJAok8&feature=youtu.be.

doing their due-dilligence to ensure that their clients are kept as safe as possible from outside threats; lest their customer base could be affected by a breach of the trust that their clients are safe.