CS808 Computer Security Fundamentals
Coursework Examination

Lewis Britton

Academic Year 2021/2022

Table of Contents

Topics

- Cryptography [1]
- Steganography & Authentication [2]
- Malware [2]
- Human Security [N/A]
- Network Attacks [1]
- Network Defense [1]
- Threat Modelling [1]

1 Cryptography

1.1 Scenario

Your friend Miriam (who lives nextdoor) has decided she'd like to share a file with you, but only with you. She's decided to encrypt the file before sending it to you.

1.2 Question

What type of encryption (symmetric or asymmetric) should she use? Given your choice, propose a specific algorithm and explain to her in a high level how it works. Miriam also doesn't know how Diffie-Hellman relates to public key encryption, explain how they're related.

1.3 Answer

1.3.1 Cryptography Background

Miriam wishes to use cryptography to store and transfer data in a secure manner to ensure that only the intended recipient has access to the data shared. This requires the data she wishes to send to be encoded in a form which cannot be easily interpreted by a human out-of-the-know. For this to work effectively, Miriam must encrypt her plain text data to the form of 'cipher text', produce a suitable key (or set of keys), and enable use of the key(s) to access the plain text data from the cipher text, given the desired recipient.

The two primary types of encryption are 'symmetric' and 'asymmetric'. Symmetric encryption is the traditional and most basic form. It utilizes the standard process of plain text → cipher text (encryption using a key) → plain text (decryption using a key), where the keys used by the sender and recipient are the same. This means that only the sender and recipient should know the key (number, string, etc.). If course, this implies a danger in the sense that sharing a common key to encrypt and decrypt can lead to an increased chance of data infiltration.

Alternatively, Miriam could use asymmetric encryption, which is also referred to as 'public key' encryption. It's a more recently developed, more secure version of symmetric encryption which uses two different keys for encryption and decryption. Much like the former, it still followed the standard framework of plain text → cipher text

$\rightarrow$ plain text however, encryption is executed using a ‘public key’ and decryption is executed using a ‘private key’. The public key is made publicly available and can be used by anyone who wishes send a message. The private key is securely sent to only recipients who are authorized to access (decrypt) data. Therefore, data which has been encrypted using a public key can only be decrypted with a private key, by the intended recipient. Intuitively then, a message which has been encrypted using a private key, generally by the intended recipient of the initial data, can be decrypted by anyone with the public key. Thus overall, this method is a more precise and secure method of data transfer.

1.3.2 Cryptography Choice

So forth, using this logic, Miriam would be relatively safe using symmetric encryption, primarily due to the scale and scope of her data transfer; she wishes to send a single file to a location she is familiar with. As she wishes to only share a file with one person and there is no mential of open encrypted replies etc., there is not much need for a public key. By the sound of it, it’s a one-way system where data must simply be encrypted and decrypted once. Thus, exchange of the single private key should theoretically not be incredibly dangerous or exposed.

1.3.3 Algorithm Background

There are various methods of encryption asscoiated with both symmetric and asymmetric types. Within symmetric encryption we generally refer to to ‘block cipher’ methods such as [1.1] Advanced Encryption Standard (AES) and [1.2] Blowfish; and ‘stream cipher methods’ such as [1.3] One-Time Pad. Note that block ciphers account for cryptographic algorithms which use fixed-length bit groups (blocks) and stream ciphers account for cryptographic algorithms which encrypt each digit of plain text one-by-one using a ‘cipher digit stream’; a ‘stream’ of digits which correspond with each digit in the plain text being encrypted. Within asymmetric encryption we tend to refer to methods such as the [3] Rivest-Shamir-Adleman (RSA) algorithm. In this context [3] Diffie-Hellmen also has significant relevance.

AES [1.1] is a symmetric key algorithm which means it uses the same private key for encrypting and decrypting data. It generally followed a fixed block size of 128 bits with an associated key size of either 128, 192 or 256 bits. AES operates on a column-major matrix, with an associated key size relative to the number of ‘rounds’ required to process the plain text to cipher text; with 10, 12 and 14 rounds for 128, 192, and 256 bit keys, respectively. For example, accounting for 128 bits:

$$\begin{bmatrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \end{bmatrix}$$

AES involves numerous intricate steps in order to be effective. [1] the 'AES schedule' must be used to produce 'round keys' from the cipher key. Thus, $n + 1$ 128 bit round key blocks are needed relative to n rounds. [2] each byte of the matrix is combined with a byte from the round key using XOR. [3] each byte is replaced by anohter byte from a matrix called the 'forward S-box'. [4] matrix algebra: transpose takes place during which the final three rows of the matrix are shifted relative to the substitution. [5] step 2 is executed.

Blowfish [1.2] is a symmetric key algorithm which means it uses the same private key for encrypting and decrypting data. Much like AES, it was designed to be an alternative to the Data Encryption Standard (DES). It generally follows a 64 bit block with key blocks ranging from 32–448 bits; is sixteen rounds and like AES, relies on S-boxes. Basic exmaple tend to consist of five arrays including a P-array (K) and four S-boxes (S (F-function)) $\forall S\{0, ..., 3\}$, for r rounds. Data from the plain text is separated into left (L) and right (R) sections. So forth, for each round in the process, [1] L is XOR'd with r^{th} P-array; [2] XOR'd output carried to F-function; [3] F-function output is XOR'd with R; and [4] $L \leftrightarrow R$. Within the F-function, there consists a process which segments the S-boxes, mod them, and XOR them to produce a cumulative product.

One-Time Pad [1.3] is a symmetric key algorithm which means it uses the same private key for encrypting and decrypting data. Unlike block cipher techniques, this stream cipher one uses a one-time use key which tends to be of identical size as the plain text being encrypted. Simply, plain text is assigned a random private key and every bit of the plain text is separately encrypted using each relative bit in the assigned private key. This is executed through modulus algebra. Provided that the generated private key is completely independent of the plain text, it's randomly generated, the same size as the plain text, and it's completely original and unique; the One-Time Pad method should hold to create secure cipher text upon each independent occasion.

The RSA [2] algorithm is an asymmetric key algorithm which means it uses different (public and private) keys for encrypting and decrypting data. This method is simple but very effective. The first step is to select to prime numbers (x and y) which are so immense that they are unguessable to an average human. And then, compute n for x and y, a.k.a.:

$$n = xy$$

Next the function of the totient (the N numbers $< n$, which share no common factors with n) is calculated:

$$\phi(n) = (x - 1)(y - 1)$$

In the following step, a unique separate integer, call this i, is selected such that:

$$i \xrightarrow[co-prime]{} \phi(n)$$

and:

$$1 < i < \phi(n)$$

In result, (n, i) forms the public key. The next step is to compute a new value d for:

$$id = 1 mod \phi(n)$$

So forth, (n, i) is the public key and (n, d) is the private key; the keys have been generated. Furthermore, to encrypt given unencrypted plain text P_u using public key (n, i), and produce cipher text C; the cipher text is calculated:

$$C = P_u^i mod n$$

and, cipher text C is decrypted to decrypted plain text P_d using private key (n, d), as calculated:

$$P_d = C^d mod n$$

Note that this is particularly secure as it is impractical to compute d from only n and i, leaving no unathorized access to the private key from the public key.

Finally, the Diffie-Hellman key exchange [3] is a method of 'securely' exchanging private symmetric keys over a public network. Through this, an attacker or interceptor should not be able to access or derive the key(s) from any data given. The fact that the two parties may be sharing keys across a public network implies that they may not have any prior knowledge or experience of one-another and thus, the exchange may also be made over an 'insecure channel'.

A two-party (A and B) example best highlights the structure of the Diffie-Hellman technique. The parties begin by selecting a common, arbitrary number which is shared and it does not matter if this number is comprimised, although it should vary upon each execution. Additionally, each party also chooses a different number which they keep secret. Each secret number is then mixed with the common number resulting in two new unique numbers which can then be publicly exchanged as the common number is masking both of the secret numbers. Thus, when each party recieves the mixed nuber of the other, they can then mix this with thier own secret number to produce a final common product. These of course, will be identical across parties.

The former uses layman terms to describe the Diffie-Hellman key exchange however, the easiest way to understand it is in alebraic implementation. So forth, two parties agree to a prime modulus p and prime root modulo (every number $\{a\}$ co-prime to n is congruent to power $gmodulon$) g. This sees that the final product can take any value ranging between $1 \rightarrow (p-1)$. Next, each party selects their secret numbers a and b, respectively and A sends $A = g^a modp$ to B. Thus, B sends $B = g^b modp$ to A. A then calculates $s_A = B^a modp = c$ using their own secret number a, and B calculates $s_B = A^b modp = c$ using their own secret number b. Hence, a, b, s_A and s_B are private, and c is the final shared message but privately derived using different numbers but the same technique; so $s_A = s_B$. p, g, $g^a modp$, and $g^b modp$ are of course public. This is incredibly secure as it is humanly, mathematically infeasible to compute $g^{ab} modp = g^{ba} modp$ given the publically available data. This result c is therefore, then used as the shared symmetric private key.

1.3.4 Algorithm Choice

Given the security implications and the fact that Miriam will be carrying out symmetric cryptography with her other party, the Diffie-Hellman key exchange is a suitable method for her to practice also.

2 Steganography & Authentication

2.1 Scenario 1

Your friend is discovering steganography for the first time and wants to try out software which hides data in digital images. They are not certain which image (1 (monotone) or 2 (color-dense)) would be best.

2.2 Question 1

What criteria should be considered in making this decision. If hiding a large file, which image would be best? Would it differ if hiding a small file?

2.3 Answer

2.3.1 Steganography Background

Steganography is a type of data masking which is generally used by people who wish to hide shared information in an easily trnsferrable manner. Thus in steganography, this is known as hiding data 'in plain sight'. This often is carried out by hiding data in multimedia transfers, hence the literal meaning of steganography: *steganos* - covered; *graphie* - writing. It is particularly popular in the modern day due to the great use of mobile business machines and handheld devices.

The porcess of steganography includes three critical components: [1] the 'cover object' - the object (the multimedia) that you wish to hide data in; [2] the 'payload' - the data (object) you wish to hide and; [3] the 'stego-object' - the altered version of the *cover object* that now contains the *payload*. It's said that the human eye should be indifferent to sight of *stego-object* and the original *cover object* to best disguise the *payload*. That is for example, when using the Least Significant Bit (LSB) approach, an edited pixel of an image should be so insignificant that it looks identical to the original, to a human eye. Unlike cryptography, steganography doesn't hide the existence of a message (*payload*), it disguises it in already existing content. This could be argued to decrease suspicion surrounding message secracy. Beacuse this form of security uses already existing content, it's also very cheap and accessible to users in-the-know.

2.3.2 Least Significant Bit (LSB) Approach

The general steganography framework of multimedia embeddedness covers transfer in the form of [1] text, [2] imagery, [3] video, [4] audio and, [5] networks. Which, when using the LSB approach, all involve generally the same technique of reconfigurating data to minorly alter the properties of multimedia. Using imagery (a digital uncompressed image) as an example in the context of LSB, alteration is commonly made to the final bits in 8bit (byte) binary sequences of pixels to embed a message. The final pixel is known as the 'least significant bit' as altering it has the smallest effect on the visual appearance of the image post-*payload*-addition. That is, the *stego-object* is not distinguishable from the *cover obect*, to the human eye. This is why high-contrast, color-varying images are most effective for this; larger changes are less noticable when compared to more monotone images. LSB is regarded one of the most versatile and important applications of steganography today, due to its application in RGB bitmaps, and JPEG attribute frequencies, etc.

2.3.3 In Practice

Pre-Payload	**Post-Payload**
01010010	01010011
01001010	01001010
10010111	10010111
11001100	11001101
11010101	11010100
01010111	01010111
00100110	00100110
01000011	01000011

Table 2.1: LSB Example

LBS is most easily understood using an $N \times M$ grey-scale image (Picione, et. al., 2006). This is because each pixel can be represented as in 8bit binary $\forall\{0 \longrightarrow 255\}$. For example, to embed the letter 'Z' in a sample of eight pixels from an image, as Z's binary representation is 8bit, the modifications are made to the pixels as seen in Table 1.

2.3.4 Selecting An Image

Given the previously discussed, it's clear that image 2 is the correct choice for encoding text data upon. It proves most effective in the most relevant areas taking into consideration when selecting a cover image: ...

First off, although the images appear to be the same physical resolution, image 2 contains greater attributes which is beneficial in data masking. Image 1 is monotone and image 2 contains a great amount of colours. This means that small chnages in colours will be less noticable when the stego-object is produced; variations in pixels will therefore be more noticable when large series of adjacent pixels are the same/similar colors.

Furthermore, image 2 also has greater color depth, meaning the colors are more dense and once again small variations made by the payload will be less noticable.

Image 2 also has greater variation in color, in the sense that there are not only a lot of colors but these colors do not cover many pixels at a time; they vary in short intervals. This means that even if a few pixels of a certain color were significantly altered by the payload, it would be less noticable than it would be on image 1 as the variation would make the transition closer to 'seamless'.

Image 2 also previals regarding texture. On the surface, image 1 only contains two textures: a gradient, and ripple created by shading variation. On the other hand, image 2 varies texture very often; based not only on shading but on shape and consistency of colors also. As image 2 is based on organic objects, the shapes are inconsistent which is good when considering steganography. Again, this simply means that the variation in observable texture will better-mask pixel, not only color but, cumulative shape and shade variation.

2.4 Scenario 2

A hospital emergency department decides to digitise their patient records. Currently patient records are paper-based, and staff carry them around the hospital as necessary. The problem is that records are being left in rooms with patients, who can clearly see them. Also, records are being lost and are not always returned to the main storage cabinet. This means that in emergency situations medical staff are unable to access the records as quickly as they need to.

The proposal is to place a computer device in each of the common areas (such as the waiting room and reception desk), as well as in each of the cubicles where patients are dealt with. The staff who need to access the data records include administrators (who check patients in), nurses, and doctors.

An authentication mechanism needs to be selected for the devices. Consider each of the following questions, and propose a solution given an unlimited budget. You might need to do some research.

2.5 Question 2

What should you consider when selecting an authentication mechanism for this scenario? What might the requirements be? What options are available for authentication? How does each option match your requirements? Given the previous answers, which option would you choose and why? Assume now that you have a smaller budget, what impact would this have on your choice?

2.6 Answer

2.6.1 Authentication Background

Authentication is commonly considered a three-step porcess which ultimately decides if you are who you say you are, accoring to the machine to which you are attempting to gain entry. The steps include: identification - where you claim to be a user based on input data (i.e. username); authentication - where you attempt to gain unique entry relative to the user you are claiming to be (i.e. password or other types of verification / unique identifiers); and, authentication - where the machine checks you against a database of user data, uniquely identifies you and decides whether or not to grant you access. This porcess aims to ensure the deployment of secure systems and secure porcesses and transactions happing within these systems and between the systems.

2.6.2 Authentication Factors

Intuitively, authentication is the most important part of this process. On the very base level, users must assign a password with thier unique identifier, which is required upon each access attempt. This is known as single factor authentication (SFA). It is objectively the least secure form of authentication as attackers only require one acess key to infiltrate the data of victims; and this unique identifier is not actually unique relative to someone's person. Additionally, the 'password problem' is also present. This is a problem which makes passwords as a single-factor authentication method vulnerable because of factors related to the fact that a password is simply a tring of text. Many people re-use passwords for different services, meaning that if an attacker gains access to one of your accounts, they've gained access to them all (given (SFA)). Users can also be extremely näive in creating passwords in the sense that they may be easily guessable, they may be shared with other people, and they may be recorded in obvious locations, etc. These of course all make attacks easier for the penetrator. Even what we consider 'strong authentication'; a password with a great number of characters, a variation of letter cases, character types, etc., is still just a simple string.

To combat this, most bodies which require authentication have introduced either two-factor (2FA) or multi-factor (MFA) systems which accept a greater number of factors, unique to a user. Common authentication factors range from knowledge-based inputs, to biometric identifiers, and unique 'tokens'. For example, a multi-factor authentication process may involve, after username entry; a password, an iris scan, and a token acceptance on a different device belonging to the user - connected to the same service, etc. thorughout any authentication process, details which are inputted by the user upon access request are compared to details stored either locally or on an authorization server, on the non-user end, to determine entry. If all credentials entered by the users perfectly match the database records, they are granted entry.

2.6.3 Knowledge-Based Athentication

Knowledge-based authentication is based upon something the user knows, such as a man-made thing like thier username and password. As discussed, this factor along is not very secure.

2.6.4 Posession-Based Authentication

Posession-based authentication relates to something the user own by nature. This is most commonly seen in the form of something man-made like a phyisical key (for more manual situations), or a hardware piece such as a computer to open their e-mail client and recieve emails. Generally these tools can be used as additional forms of authentication as they may be assigned to the user in the non-user end database and when the user uses this tool, there is notification on the non-user end, telling them that the user has accepted thier request to verify themselves. A commonly used example in this case is signing in to another associated device to retreive and confirm a One-Time Password (OTP).

2.6.5 Inherence / Biometrics-Based Authentication

Generally, this factor is considered something which accounts for what the user 'is'. That is, a human factor. It accounts for factors which are naturally unique to a human being, which can be captured, stored, and compared upon future request. This is also referred to a biometric authentication. This tends to be segmented into physical and behavioural categories. Common biometrics often include features such as finger/thumb print scanning, facial recognition, retina scanning, etc. These factors are often used in conjunction with other authentication (multi-factor) methods to assist the matching of human identification to knowledge-based access. For example, in mobile authentication on an application, or as part of API authentication. This is where an HTTP server requests one form of authentication, as standard, and an API key is generated and assigned to the user, then requested on an associated system upon acess attempt. A similar form of this is Open Authorization (OAuth), where tokens are generated to allow thrid parties to access necessary user data without access to passwords or other authenication data. This is common in 'linked accounts' etc.

Though biometrics can be extremely effective in distinguishsing individuals when it works flawlessly, it is still digital which means failures are common. This can lead to four possible outcomes which can be quantified:

True Accept	*Correct* user *granted* access
True Reject	*False* user *denied* access
False Accept	*False* user *granted* access
False Reject	*Correct* user *denied* access

One of the most commonly applied forms of physical biometrics in the modern day is facial recognition. It reads the contours and unique geometry of the faces of users and compares them to data on record, often implemented to allow users seamless access to their systems or devices. This method does however, carry a great deal of false accepts, or fale identifications. Particularly in circumstances where facial recognition is critical in determining criminality etc. This causes huge, uneccessary legal problems.

Iris scanning is another popular application. It's the simplest form of eye scanning and can actually be fairly accurate time-after-time as iris patterns differ greatly between humans. Iris patterns can be easily stored as they can be represented as 2D vectors. Ifrared cameras scan and compare these patterns upon humans and compare to databases. Again, for seamless access attempts.

Retinal scanning is anoter, more advanced, form of eye scanning. This is less commonly used in commercial product such as household systems and mobile devices etc., and more common in secure systems such as government and military systems. It uses a similar 2D array as the former however, relies on more unique and more variable patterns in a more advanced location of the eye. It is based on blood vessel patterns which must be identified in low brilliance, low luminance light. This can of course by unreliable in the long-run due to alteration in the human eyes, enduced by ageing, development, disease etc. However, bodies which implement this technique usually store the capital to invest in maintenance and update of these systems.

Fingerprint scanning was one of the first forms of biometric authentication to be used in a commercial setting. In the present it's used in places anywhere from everyday device access, to two or multifactor authentication in more advanced systems; commercial, professional, and military etc. It's based on the contour patterns of the human finger. Minor alterations in these contours can cause huge differentials between human fingerprints; based on ridges, valleys, peaks, etc. As always, they're scanned and compared to records. The comparison itself is fairly reliable however, the process can become unreliable very easily if there are defects on fingerprints or the reciever.

As for behavioural biometrics, the associated authentication techniques are based on human patterns. That is, the way humans interact with machines, not out of physical nature but out of personal attributes. The most common type is keystoke pattern analysis. It's based on the key 'dwell time' - the duration of time for which a key is depressed, and the key ‘flight time’ - the elapsed time between each keystroke. This can be fairly inreliable as these patterns may vary over time relative to a user, and it also opens up opportunities for keylogging attacks, etc. If a user uses a mouse with their machine, similar patterns can be analysed in that regard. These rules can also be applied to most other hardware pieces. Although practices of this have proven to be accurate, they can be expensive to record, implement and maintain, especially in cases where these methods are used as two or multi-factor authentication.

2.6.6 Location-Based Authentication

In location-based authentication, GPS may be used to determine the location of a user attempting to gain access. This of course would never be used as a sole factor due to population scale however, it can be very useful in verifying whether or not a user is attempting to gain access from the location they claim they are. Even in circumstances where users do not have access to GPS services, they can be located using their relative network profile; even just adding that little bit of extra security on the end of a protocol. This is an intuitive factor as it stops attackers outside the regular sign-in location being able to gain unauthorized access as there is an average location log to compare sign-in locations to.

2.6.7 Time-Based Authentication

The time-based authentication factors works in a very similar manner to the location one. Although times when a user signs in to services can naturally be more variable than the locations they sign in from, there is likely to still be some form of pattern in the human habits of the user. This factor also has more scope. Times are not only compared on a linear basis (i.e. across patterns of historic access times); they’re compared on a recurring basis, often in conjunction with the location factor. For example, someone trying to gain access to a user account from Meikle Bin, ten minutes after the last sign-in which was from Meikle Dripps would be denied immediately due to the infeasibility and generally, impossibility.

2.6.8 Token-Based Authentication

In the case of he use of Hypertext Transfer Protocol *Secure* (HTTPS), the discussed would traditionally require a user to re-enter and re-affirm all authentication details upon each access request. However, ‘tokens’ have since been introduced, which are assigned to the user-end of the authentication process and are assigned relatively to the user upon each access request. This results in cases where users do not have to sign-in to applications upon each visit, given the appropriate circumstances determined by the non-user end.

2.6.9 Algorithms

Various algorithms have been developed which make the previously discussed methods more feasible in the modern day. The most commonly known set are the Secure Hash Algorithms. These range across an array of developments from the Secure Hash Algortihm 1 (SHA-1), through more recent SHA-256, SHA-384 and SHA-512. They use hash functions which accept data of variable size and length and create and assign this data to fixed values. They execute this using standard bitwise operations (analysing data on a bit-to-bit basis), modular algebra (as seen previously), and compression functions to produce the final finite product (creating fixed-length data). In result, for example, the input is a variable-length string, the hash functions are executed, the output is a fixed-length differential string. The most common use of this is password encryption. This is extremely effective as hashed output contains no attribites related to the original input.

Another form of these SHA methods is Method Digest 5 (MD5) which tends to be used to encrypt attributes of other authentication methods which are greater than passwords. They operate in a very similar manner however; deriving fixed-length data through hash functions.

As for methods such as facial recognition, which is become more commercial as time goes on, algorithms such as Eigenface exist. Eigenface accounts for the facial attributes related to unique identifiers in the contours of the human face. It constructs a *cov* matrix using baseline 'eigenface' images which allows smaller derivations to be made and compared to on a one-by-one basis. It actually works in a very similar way to the famous Fama and French (1970) K-factor investment model.

2.6.10 Recommendations

In these circumstances, it's clear that the hospital require a secure multi-factor authentication system which can be used extremely quickly and seamlessly, as doctors etc. will likely require access in perhaps intense or busy moments. The best solution to this would be a database system which involves three-step access. Using the SHA hashing system on passwords as a standard practice keeps level-one access secure and reliable.

On level-two, provided the time is taken to initiate and record data required, facial recognition should be used to ensure only those who should be viewing patient data, are. Fingerprint scanning would be a reliable level-three method.

Furthermore, admins, doctors and nurses all require access to the database for different reasons and on with varying levels of urgency. This means that upon entry, users who tend to work and idle in the relevant databases for longer periods, such as admins and nurses should generally be required to use the three-level authentication system upon each entry request. However, in situations where doctors are dealing with patients and may be operating on a much faster-paced workflow, they may only be required to use one form of authentication such as the facial recognition or fingerprint scan level to make the process more seamless. This can be kept secure by the use of tokens which account for the doctor's first full three-step login. This could be on a time-basis and require the doctore to continue authentication after finite periods.

3 Malware

3.1 Scenario 1

John discovered a virus on his office computer. The virus expert was called and took half a day to clean the computer and recover the data. The following day, the same virus came back. After spending several days fighting the virus, it was discovered that John himself unkowingly infected his computer immediately after each cleaning.

He had a game that he liked to play during his lunch break. His wife, a student, brought the game from college on an infected USB flash drive. Every time John inserted the drive into his office computer, the virus installed itself afresh.

3.2 Question 1

What should the company policy be?

3.3 Answer

The term ‘malware’ refers to any malicious software which is intended to be present upon and inflict harm upon a computer system. Primarily, it may either be designed to terminate processes on a machine, or return data from said machine to the creator/sender. These intensions could be driven by the cretor’s financial benefit, the creator’s satisfaction, political conflict, reason to stall the reciever, etc. As malicious software can vary greatly in intension and purpose, there are many ways in which a user may be able to identify if there is malicious software present on their system. Although, like any human illness, there are not always signs. Some common symptommes include: reduction in optimization of system hardware and rescources, resulting in slowed performance. During this, CPU and memory usage may be abnormally high, internals fans may be running at full speed, etc. Furthermore, sympotommes such as local ads, crashes, unwanted programs being installed, loss of browser control, etc., may also become present. For reference, most common types of malware include: viruses, worms, trojans (trojan ‘horses’), adware, spyware, ransomware, rootkitting, and keylogging.

3.3.1 Viruses

A virus is a picee of malware which attaches itself to a ‘host’, i.e. another program. When the user of the infected machine interacts with the host, the virus is unleashed and executed. The virus then replicates itself across the infected machine onto child hosts, seeking to infect as many things as possible. This is often refered to as 'mutating'. Much like a serious human plague. A virus always leaves a digital signature at each host it has infected.Viruses are very commonly introduced to a machine, especialy by less knowledgeable individuals. They are often contained in emails (may relate to phishing or spyware in an attempt to maliciously obtain user data) or downloaded from untrustworthy websites (may relate to spyware and keylogging). More specific viruses may operate on a 'time bomb' or 'logic bomb' basis, where they execute upon their host at a specific time or when a specific set of criteria are met, respectively. In summary, viruses *conceal* themselves, they *propigate* and spread across hosts, and they leave a *signature* at each host. Their purpose is to remain under-the-radar (*concealed*); execute a *payload* which may either hold user data to ransom (ransomware), or monitor user data which may be used maliciously (spyware); finally, the viruses *propigation* ensures it covers as much ground as possible to access/obtain as much data as possbile.

3.3.2 Worms

Worms contain some similar properties to those of viruses however, they do not requre a host to attach to. Therefore, worms dont need a user to interact with them for them to be activated. They still act in the capacity of a self-replicating infection (*propigation*) with the intent to infect as many things as possible on a machine or a network. As they don't require user interaction, they can spread across systems and networks a lot faster than virsus, potentially making them more dangerous, especially to those out-of-the-know. Because of this, worms are often used in loops across networks in an attempt to create Denial of Service (DoS).

3.3.3 Trojans

A malicious trojan operates in much the same manner as a traditional trojan horse; it appears to be harmless with the hopes that the user will be accepting of it and once it gains access, it unleashes harm. Much of the time, trojans do actually complete the harmless tasks which the user believed it to be intended for - to create a greater disguise. This is known as 'back door' access. Due to this, it is often considered as one of the most dangerous types of malware, especially as users out-of-the-know may never be able to identify the source of their problem. They can be disguised physically or on a machine. For example, they may be disguised as a harmless piece of software available for download, or as a labelled external drive: 'student grades', etc. As trojans require user permissions, they are often used to gain more specific data such as personal data, financial details, or hold data ransom (ransomware).

3.3.4 Intensions & Sub-Types

Adware is usually introduced to a system by a virus. It is generally unwanted, irrelevant software which displays an abnormal amount of advertisements, either locally using some maliciously installed program, or within a web browser. It is often either intended as a program to entice a user to click on (activate) a link in the ads which may install further malicious software, or as a forced promotional tool either for the creators financial benefit or as a form of spyware intended to gain user data.

Furthermore, **spyware** is simply malicious software which is intended to maliciously gain access to and record user data which is stored on a system or network, and return it to the creator.

Likewise, **ransomware** is also intended to gain personal user data however, is executed over a more strategic method. Once executed, through infiltration from some form virus or worm etc., it prohibits a user from being able to use the infected device by using either proprietary methods to the device or by encrypting files. The user them must make some form of exchange, usually in the form of personal data/information or cash, in order to regain access to their device. In more recent times, cryptocurrency miners have been heavily targeted, with the attackers forcing them to make enormous crypto exchanges.

Finally, **rootkitting** and **keylogging** are both popular methods of maliciously obtaining and practicing administritive actions on a system or network. A 'rootkit' allows an attacker to virtually interact with a system like it is their own, thus gaining access to any data they wish. Similarly, 'keyloggers' maliciously track the keystrokes of users and returns them to the attacker. Patterns are analysed in search of sensitive data such as passwords etc., in reuslt giving attackers advanced access and permissions on the system of the attacked user.

3.3.5 Company Policy Recommendations

First off, this virus may have been infliced upon John's office computer deliberately by his wife, on the basis of trojan methods. She may have been aware that the flash drive which contained the game she gave to John contaned malicious software and therefore, had disguised it verbally as something leisurely which she knew John would be interested in.

Of course, John should have been more aware of the dangers of inserting an external drive, which he didn't create the contents of, into any device however, there are two issues here which should also be tackled on a corporate level. The first company policy amendment, assuming it's not already in place, is strictly no games or leisurely activity on office systems. This should be a given as even if games aren't installed from external devices, doing anything unknown like this could introduce viruses and worms etc., which could potentially infect the entire company network. Secondly, it should also be company policy to not permit any interfacing of external devices with corporate machines. That is, regardless of whether the contents of a device are known to / created by the user, there should be no interaction between corporate software and software contained on an external device. This simply protects the corporate network agaisnt any intentional malware in the case where not all contents of the external device are known / the device has come from an unknown source; or external human error in the case where the contents of an external drive are created by a corporate machine user.

3.4 Scenario 2

You've discovered malware on your system. After analysing it, you notice it behaves as follows. After infecting your machine by attaching itself to a file, every Monday a message appears on your screen at 9am with a picture of a cat called Garfield saying 'I hate Mondays'. It sends itself out to all your mail contacts.

3.5 Question 2

What type of malware is this and why? What could the impact of this malware be if it were on an organisation's network? What is the trigger for this malware and how should you classify it? What is an example of an alternative trigger?

3.6 Answer

The term 'malware' refers to any malicious software which is intended to be present upon and inflict harm upon a computer system. Primarily, it may either be designed to terminate processes on a machine, or return data from said machine to the creator/sender. These intensions could be driven by the cretor's financial benefit, the creator's satisfaction, political conflict, reason to stall the reciever, etc. As malicious software can vary greatly in intension and purpose, there are many ways in which a user may be able to identify if there is malicious software present on their system. Although, like any human illness, there are not always signs. Some common symptommes include: reduction in optimization of system hardware and rescources, resulting in slowed performance. During this, CPU and memory usage may be abnormally high, internals fans may be running at full speed, etc. Furthermore, sympotommes such as local ads, crashes, unwanted programs being installed, loss of browser control, etc., may also become present. For reference, most common types of malware include: viruses, worms, trojans (trojan 'horses'), adware, spyware, ransomware, rootkitting, and keylogging.

3.6.1 Viruses

A virus is a picee of malware which attaches itself to a 'host', i.e. another program. When the user of the infected machine interacts with the host, the virus is unleashed and executed. The virus then replicates itself across the infected machine onto child hosts, seeking to infect as many things as possible. This is often refered to as 'mutating'. Much like a serious human plague. A virus always leaves a digital signature at each host it has infected. Viruses are very commonly introduced to a machine, especialy by less knowledgeable individuals. They are often contained in emails (may relate to phishing or spyware in an attempt to maliciously obtain user data) or downloaded from untrustworthy websites (may relate to spyware and keylogging). More specific viruses may operate on a 'time bomb' or 'logic bomb' basis, where they execute upon their host at a specific time or when a specific set of criteria are met, respectively. In summary, viruses *conceal* themselves, they *propigate* and spread across hosts, and they leave a *signature* at each host. Their purpose is to remain under-the-radar (*concealed*); execute a *payload* which may either hold user data to ransom (ransomware), or monitor user data which may be used maliciously (spyware); finally, the viruses *propigation* ensures it covers as much ground as possible to access/obtain as much data as possbile.

3.6.2 Worms

Worms contain some similar properties to those of viruses however, they do not requre a host to attach to. Therefore, worms dont need a user to interact with them for them to be activated. They still act in the capacity of a self-replicating infection (*propigation*) with the intent to infect as many things as possible on a machine or a network. As they don't require user interaction, they can spread across systems and networks a lot faster than virsus, potentially making them more dangerous, especially to those out-of-the-know. Because of this, worms are often used in loops across networks in an attempt to create Denial of Service (DoS).

3.6.3 Trojans

A malicious trojan operates in much the same manner as a traditional trojan horse; it appears to be harmless with the hopes that the user will be accepting of it and once it gains access, it unleashes harm. Much of the time, trojans do actually complete the harmless tasks which the user believed it to be intended for - to create a greater disguise. This is known as 'back door' access. Due to this, it is often considered as one of the most dangerous types of malware, especially as users out-of-the-know may never be able to identify the source of their problem. They can be disguised physically or on a machine. For example, they may be disguised as a harmless piece of software available for download, or as a labelled external drive: 'student grades', etc. As trojans require user permissions, they are often used to gain more specific data such as personal data, financial details, or hold data ransom (ransomware).

3.6.4 Intensions & Sub-Types

Adware is usually introduced to a system by a virus. It is generally unwanted, irrelevant software which displays an abnormal amount of advertisements, either locally using some maliciously installed program, or within a web browser. It is often either intended as a program to entice a user to click on (activate) a link in the ads which may install further malicious software, or as a forced promotional tool either for the creators financial benefit or as a form of spyware intended to gain user data.

Furthermore, **spyware** is simply malicious software which is intended to maliciously gain access to and record user data which is stored on a system or network, and return it to the creator.

Likewise, **ransomware** is also intended to gain personal user data however, is executed over a more strategic method. Once executed, through infiltration from some form virus or worm etc., it prohibits a user from being able to use the infected device by using either proprietary methods to the device or by encrypting files. The user them must make some form of exchange, usually in the form of personal data/information or cash, in order to regain access to their device. In more recent times, cryptocurrency miners have been heavily targeted, with the attackers forcing them to make enormous crypto exchanges.

Finally, **rootkitting** and **keylogging** are both popular methods of maliciously obtaining and practicing administritive actions on a system or network. A 'rootkit' allows an attacker to virtually interact with a system like it is their own, thus gaining access to any data they wish. Similarly, 'keyloggers' maliciously track the keystrokes of users and returns them to the attacker. Patterns are analysed in search of sensitive data such as passwords etc., in reuslt giving attackers advanced access and permissions on the system of the attacked user.

3.6.5 Relative Malware & Impact

It is clear that this piece of malware is a virus which has been likely inflicted upon the system by the user through interaction with a malicious e-mail or webpage. The process of it attaching itself to a file is known as *mutation* when a virus finds its way locally onto a machine and finds a *host*. This is especially dangerous on a corporate machine or network as when a virus *propigates* across hosts, it could possible infect and damage all machines on the network. As this virus specifically executes upon 9am every day, it's clear that it operates on a *time bomb* trigger, where specific protocol is set to occur at a pre-determined time of day. A siliar trigger to this is a *logic* trigger, where the protocol is set to occur when a specific set of criteria are met. Such as a log-in.

4 Human Security

N/A

5 Network Attacks

5.1 Scenario

Your friend Alex has decided to start an online business selling digital art they produce. To do this Alex has purchased a machine which they have configured as a web server. The web server is stored in an office which he rents. In the office there is also a computer which is connected to the LAN and the internet.

Alex often works from home. At home, Alex uses a personal computer and a tablet which are connected to their LAN using a hub. Alex also has a router which allows him to connect to the internet. Work Alex completes a PC is stored on the local HDD and is uploaded to the web server from the HDD when necessary, through the internet. From the office Alex can also upload any work to the web server on the office LAN.

5.2 Question

Analyse the potential threats to security Alex currently faces based on the material presented in the module.

5.3 Answer

5.3.1 Background

A computer network is an intangible which can aid the linking and communication of devices which are not physically related or connected. This communication is most often used to share and exchange resources. For example, the original efficiency-enhancing purpose of netowrks was to make corporate electronic mailing and telecommunications more accessible and effective. Of course, this is where we see the first potential gap for authorization issues. That is, there are always intended recipients of data shared across networks however, network attackers often see that the complete process is not fulfilled.

5.3.2 Comminications & Interfaces

To understand the various ways in which attackers can intercept network comunications, it's important to observe network communication types. The most frequent network distributions are: Local Area Networks (LAN), which span across a specific area such as an office; Metropolitan Area Networks (MAN), which span across primary urban city landscapes; and Wide Area Networks (WAN), which span across major geographical regions such as countys and boroughs. Networks operate across *network links* which use physical mediums such as electrical and optical fiber cables. The most common application of electrical and fiber data transfer in LAN, MAN and WAN distributions is *Ethernet*.

Many systems in the modern day now utilize wireless interaction known as Wireless Local Area Network (WLAN); often referred to as 'WiFi'. This method is not nearly as effective as traditional interfacing however, often leads people to believe they are communicating more efficiently. This communication relies frequently on radio communication across cellular networks using communication satellites.

To communicate data across a network, Internet Protocol (IP) transmissions in which data is packaged into what's referred to as 'data packets'. These are then sent along a pre-determined network path. Start and end points are identified for the data transfer, which are referred to as 'ports'. Ports are used to filter different types of data into the correct streams so they end up at the correct type of recipient. These port protocols are then associated with with unique IP addesses to identify individual recipient systems. The most commonly used ports on the World Wide Web (WWW) are of course 80 and 443 for HTTP and HTTPS.

To allow machines the communicate across networks, there are various hardware requirements which must be met in order to give a machine the necessary capabilities to do so. A machine must use some form of physical network interface internally such as a Network Interface Card (NIC). These allow physical Ethernet connection to a machine and transfer data via electrical signals along the line. Features of a NIC are often also integrated in modern motherboards however, there's of course no more efficient mehtod than using a traditional NIC. Network Hubs and Switches may also be introduced in an environment to allow multiple machines to connect to one place (a network) and then transfer their data. Additionally, 'routers' are available to those who wish to gain access to the internet, and support data transfer to and from the internet. Modern routers also support wireless connections however, they are often unreliable.

5.3.3 Server Vulnerabilities

Issues can occur at various points in data storage and communication on networds. Data may be left in a particular file system or may be cached which opens them to other users who may not have had original access or may not be intended as target recipients. Much of the data held locally on machines connected to a network may not utilize proper encryption protocols, likewise with data being communicated across a network. This again creates an opportunity for attackers.

More major-scale server vulnerabilities may be indicated by a Denial of Service (DoS) attack. In these attacks, the attackers aim to make a network temporarily unavilable to its regular users, frequently achieved by barraging a server with an unmanageable quantity of requests. This is often in an attempt to create data vulnerabilities on the server-end. Additionally, Distributed Denial of Service (DDoS) attacks follow the same framework as DoS attacks however the source of the barrage is spread across a great number of locations. This simply makes it more difficult for the target to identify the source and combat the attack.

5.3.4 Port & Router Vulnerabilities

It's important to note that ports can take on various states: open, closed, filtered and unfiltered. Leaving ports open may create vulnerabilities so conducting a port scan can highlights particular ports which may be leaving gaps for attacks. Generally, the protocol is to close ports if a scan indicates that they aren't serving any crucial purpose for the network.

Furthermore, more physical issue become present when routers and other systems alike are left unattended and unmontiored. That is, just as ports can be left 'open' and vulnerable to attacks, so too can physical interfaces. Of course in an event of physical interception, an attacker must be in the presence of the system. If they are, they may interface potentially dangerous connections with the network outlet which could be used for a variety of attacks or data redistributions etc.

5.3.5 Communication Vulnerabilities

Recall 'packets' being sent across a network. This has lead to a vulnerability in this communication form known as 'packet sniffing'. This is fairly self-explanatory as it refers to a program which intercepts or logs a packet and/or other traffic in transit across a network.

Additionally, a Man(Machine)-in-the-Middle (MITM) attack may occur, which involves an attacker making an attempt to alter the communication process between a sender and reciever of data on a network. In effect, the attacker acts as a third entity making themselves so it appears as though there remains communication only between two entities. The attacker does this by controlling, selecting and relaying communication messages between the two original entities. The attacker also often manufactures new communications.

'Spoofing' is another example of a communication disguise. Instead of acting as an existing participant in data communication, the attacker acts as an external trustworthy source. Thus, this form of attack can be more widely implemented in places such as electronic mail, telecommunicaitons, and web pages etc. Therefore, this method, like most others, can be used to unrightfully gain access to data and information and spread malware.

5.3.6 Recommendations

Alex could potentially face any of the discussed issues as he is using interfaces which communicate with a variety of different network components.

6 Network Defense

6.1 Scenario

Your friend Alex has decided to start an online business selling digital art they produce. To do this Alex has purchased a machine which they have configured as a web server. The web server is stored in an office which he rents. In the office there is also a computer which is connected to the LAN and the internet.

Alex often works from home. At home, Alex uses a personal computer and a tablet which are connected to their LAN using a hub. Alex also has a router which allows him to connect to the internet. Work Alex completes a PC is stored on the local HDD and is uploaded to the web server from the HDD when necessary, through the internet. From the office Alex can also upload any work to the web server on the office LAN.

6.2 Question

Propose countermeasures which can be implemented to minimise the risk of attacks identified in the network vulnerabilities week which relate to this scenario. If you need to make assumptions, document them in your answer.

6.3 Answer

6.3.1 Background

A computer network is an intangible which can aid the linking and communication of devices which are not physically related or connected. This communication is most often used to share and exchange resources. For example, the original efficiency-enhancing purpose of netowrks was to make corporate electronic mailing and telecommunications more accessible and effective. Of course, this is where we see the first potential gap for authorization issues. That is, there are always intended recipients of data shared across networks however, network attackers often see that the complete process is not fulfilled.

6.3.2 Firewalls

Firewalls are one of the most commonly practiced and understood protocols which can be applied to combat attackers infiltrating a network. Firewalls monitor and filter traffic between ends in a network, based upon pre-determined criteria by users of the network. This generally takes the form of denying access which hasn't been specifically determined as safe by the network users (white list rejection), and denying access which has been specifically determined as harmful by the network users (black list rejection). They began in the 1980's as basic packet filters, as previously discussed, however have more recently grown into something more powerful and more widely applied.

Often Firewalls are applied between networks such as Local Area Networks (LANs) and Wide Area Networks (WANs). Their purpose in this case is to protect all devices connected to the relative local network and ensure no harmful breach from outwith. In the modern day, they are applied extremely widely, prividing securities such as: basic intrusion prevension, 'cloud' computing access monitoring, and identity access monitoring, etc.

As *packet filters*, Firewalls are generally applied over smaller netorks in which they analyze small amounts of traffic and data between intra-network communications, as determined by the network administrator. As a *proxy service*, Firewalls are applied over a larger scale, usually a network security system etc., which implements some of the more advanced features discussed in order to prevent any harmful data, requests or attacks entering the local/private network from the internet, for example. In the case of the use of a *proxy server*, the only device recognised by the internet is the proxy server, thus making a local network operating under this server more private; with hidden IP addresses etc.

6.3.3 Virtual Private Networks

Virtual Private Networks (VPNs) are generally implemented when using public networks in order to create a 'virtual' shell of privacy. They operate by encrypting data in network traffic and again, disguising a user's network identity. Thus, makign it more difficult to apply monitoring protocol to network users and their activity. A VPN masks an IP address by relaying it through a remote server, which means the VPN becomes the host of your activity when interacting with a relevant network. In effect, this leads to privacy in the sence that network activity, such as internet browsing, and associated data communication is not visible to external sources such as other users (potential attackers) and your Internet Service Provider (ISP). If any of these externals did gain access to user data, it wouldn't be easily interpreted and may even be valueless due to the VPNs scrambling.

Without some form of network data encryption, anyone on the network with the correct know-how could potentially access and browse a user's data. A VPN can take various forms in this case. The most common undertsanding of a VPN is *secure encryption* where all of your data is encrypten when interacting with an external/public network which means without an encryption/decryption key, it would take attackers an unjustifiable length of time to gain access to user data using methods such as a 'Brute Force Attack'.

VPNs also act effectively as a *location scrambler* and *remote content access horse.* As VPNs act like a proxy for users and their associated location data is based on servers in irrelevant locations (to attackers), the precise location of a VPN user cannot actually be determined very easily. Along these lines also, the use of a VPN can allow user access to a wider scope of regional content as VPN servers act on local servers which means users can pass between servers; effectvely changing their location. Given the use of *VPN location spoofing.*

Additionally, VPNs support *secure data transfer* in the sense that a user can gain secure remote access to their relevant network given the correct use. For example, a user may wish to gain access to a corporate network from a remote location. Many of the discussed VPN fatures make it possible, given a VPN connection, for the users to utilize their network in this regard with the correct data encryption and authentication applications implemented by the VPN, relative to the network administrator's criteria.

VPNs are commonly applied in two different manners: the first is between local users and the internet, for exmaple, where a VPN server and a series of Firewalls work in harmony to filter malware and other malicious activity so it doesn't make its way back to the users; the second is the same framework however, mirrored between internet sites, with two VPN servers and two series' of Firewalls on either end of the communication. This method ensures there is no cross-site contamination and malicious attacks.

6.3.4 Demilitarized Zone

A Demilitarized Zone (DMZ Network) once again acts an an intermediary between the loca/private network and an external network from which there may be harmful or malicious traffic. This is most common in the modern day between private corporate networks and the internet, to ensure no attackers gain access to confidential data or protocol held within the private network. Transacting data which was intended for external use can also be done securely using DMZ, where protocol such as electronic mail, File Transfer Protocol (FTP), and Voice over Internet Protocol (VoIP) etc., can all be accessed securely from the internet without intrusion of the relevant private network.

This method utilizes separate servers (from the local private network) to host the public aspects, features and requirements of the network. This is often referred to as the 'two firewall architecture'. It uses an internal Firewall on the local priavte server end between it and the DMZ to allow secure and comprehensive access. On the other end, between the DMZ and the larger public network, such as the internet, is another Firewall to monitor and filter traffic to and from the DMZ and internet to ensure no malware or other malicious intent breaches. This protocol means that an attacker must be extremely advanced in the regard where they must breach a secure series of Firewalls, combat DMZ protocol and then breach the local pivate network, to do any significant damage to private content.

6.3.5 Intrusion Detection Systems

Intrusion Detection Systems (IDSs) once again act as monitors of traffic for identification of potential malware or harmful attack. They log all aspects of traffic and generate reports based on seurity criteria defined by the private network administrator, as part of what's known as an event management system. It is of course then the responsibility of the administrator to take evasive or combatting action. More advanced systems of this sort can respond to and act upon malicious acts; these are known as Intrusion Prevension Systems (IPSs). Although these systems are advanced, like most they cannot detect or prevent many forms of insider threats or errors made my network or system administrators which may pose harm or open holes for attack.

IDSs vary in types and relevant purpose. Network Intrusion Detection Systems (NIDSs) specifically monitor incoming traffic to the local/private network end of the line. Whereas, Host-Based Intrusion Detection Systems (HIDSs) specifically montior internal and external features of operating system files and transfer.

Furthermore, there are sub-features of IDSs which perform based on different criteria. *Signature-Based* IDSs scan for particular patterns in traffic which may resemble recorded or recognised ‘malicious intrusion sequences’. However, the issue here is that attackers are very much aware of this technique and are constantly developing new patterns or scrambling of patterns. On the other hand, *anomaly-based* IDSs implement techniques which are manufactured to catch newer-developed malware. This approach utilized features of machine learning which creates average models of secure and trusted traffic and data transactions to compare to other activity. Instead of identifying potential threats upon historical estimates, this method uses estimation based on process of elimination.

6.3.6 Digital Certificates

From another perspective, Digital Certificates exist to securitize systems based on allowing devices and alike interfaces the correct access and permissions. In effect, this ensures that local/private network users such as individuals and corporate network administrators can ensure that there are no harmful or intrusive devices attempting to join their network. This emthod can also be implemented in the context of a web browser, where websites require web pages and sites to meet a specific set of criteria to be considered authentic and safe, before presenting them and their content to a browser (physical user). This is sometimes referred to as a Secure Socket Layer (SSL) certificate.

Essentailly, Digital Certificates are collections of data which link servers with specified public keys, using Public Key Infrastructure (PKI) which accounts for a collection of entities, policy and protocol required to manufacture, distribute, maintain and revoke signatures. Digital Certificates can be issued to entites such as individuals, corporations, web entities, etc. and are fulfilled by these users using Certificate Signing Request (CSR). In this process, ecoded data accounting for the requested public key, corporate name, server use details, corporate origin, etc., is stored. This is then validated by a relevant Certification Authority (CA) who act as a securitization commission which decides whether or not a network server is of the ownership of the entity claiming so. After recieving the relevant CSRs, the CA surveys and audits the entity and grants or denies permission by ‘digitally signing’ the certificate - conveying trust from a trustworthy source.

Verification can differ depending on which type of certificate an entity opts for. The three recognized types of these certificates include Transport Layer Security (TLS), which operates upon a server such as an electronic mail or web hosting server and is implemented to ensure that the users of these services are protected using encryption. In result, a server and transact both ways using encrypted data. This protocol is generally indicated by servers sporting Hypertext Transfer Protocol Secure (HTTPS) as a coefficient to their Uniform Resource Locator (URL). The distribution of this techniqie may be [1] 'domain validated', which is the easiest and fasted-to-obtain form of certificate; [2] 'organization validated', which accounts for corporate operations and is generally applied to slightly more advanced web servers which implements features such as online stores; or [3] 'extended validation', which accounts for comprehensive authentication, primarily implemented by large corporatations who transact large amounts of private data very regularly.

Furthermore, the second type of certificate is a Code Signing Certificate. This is generally seen in circumstances where data downloaded from the internet, such as files and software etc., may be considered dangerous. Thus, the creator of this associated data must hold one of these certificates to ensure the potentials users (downloaders) that the data and the download process is secure and trustworthy. This is often not only the requirement in sole creation of data but also where data such as software in available in locations where it could be altered and made harmful or malicious by other external users.

The final certificate type is known as a Client Certificate. This is regarded an indentifier which indentifies a user to some other entity. This may be user-to-user or system-to-system etc. For example, this may be present in the context of virtual verification signatures when sending and recieving electronic mail.

6.3.7 Recommendations

Alex may choose to implement any of the securitization protocol seen above in the discussed Firewall, Virtual Private Network, Demilitarized Zone, Intrusion Detection System and Digital Certificate features.

7 Threat Modelling

7.1 Answer

7.1.1 Background

‘Threat Modelling’ in software development refers to the structured approach taken when analysing and constructing a practical solution to any possible cyber/security threats which may be present in the development, implementation or deployment of a computer software system. Threat models are generally specific to a particular context and a range of areas within that context, relative to the development purpose. Most threat models aim to forecast the scale and scope of different possibilities of ‘cyber security losses’. Many models begin by analysing historic cyber breaches of a relevant sort to the context you’re working in. From here, the common framework is usually considered and practiced, as follows: *structure* - the context/purppose of development; *identify* - the possible greay areas and opportunities for attack created by new development; *mitigate* - what can be done to help secure these areas; *validate* - what degree of success is likely to be observed in the outcome of these amendments. Two common practices are STRIDE (Sppofing, Tampering, Repudiation, Information Disclosure, Denial of Service, Elevation of Privelage) and Cyber Kill Chains.

7.1.2 STRIDE

The six components of STRIDE account for six different target properties in which threats and realistic assumptions should be considered. First off, Spoofing is a form of disguise in electronic communication such as electronic mail, telephone calls, website interaction, electronic messaging, etc., meaning that it’s primary factor of concern is *authenticity.* That is, when considering Spoofing, threat models must consider the liklihood of one human acting unrightfully on behalf of someone else or another form of entity in anticipation of malicious result. This may be in the form of e-mail phishing, for example.

Tampering is self-explanatory; it refers to one person unrightfully altering the data of someone else or another system which is irrelevant to them. This means its primary factor of concern is *integrity.* Threat models in this sense must consider the opportunities available for users or attackers to modeify data which in result may cause harm to other users. For example, in a weak university database, one may alter one-other’s exam results casuing failure where failure is not due.

Repudiation refers to unrighful denail of some form of correctness or truth. Therefore, the primary factor of concern in this context is *non-repudiation*, meaning threat models must consider opportunities for users to deny mistakes they've made which may have harmed themselves, the service provider, or other users. For example, in the real world, a hit, run and deny.

Information Disclosure refers of course to the personal and private data and information of associated users. This is obviously very much a developer-end issue which is based primarily on the concern of *confidentiality.* This means threat models must take into accound any opportunity in the development or maintenance process of a service where sensitive user data is held, used and/or accepted, and ensure there are no mishandlings. Failure to model, forecast and plan to prevent such concerns may lead to critical data leaks, internal misuse or attacks etc.

Denial of Sevice is another major issue anywhere in a modern development environment. Its concern of *availability* highlights that models must take into account any opportunities open to attackers who wish to make the service unavailable to genuine users by simple attacks such as HTTP request floods. This is a petty form of attack but can have such a great impact on an online service, especially if it relies solely on online transaction.

Escalation of Privelage refers to the user, or sometimes developer (insider threat), heirarchy and the relative permissions and access. This of course means that the primary factor of concern is *authorization.* That is, models must account for any opportunitis where attackers may gain permissions or rights relative to the system which are above their rightful distribution and allow them the possibility to easily infilct harm upon the users and the system easily, as if they are an administrator. This issue is very often the case in a corporate context. For example in legal documentation, where a user who should only be granted read permissions but gains read and write permissions, and alters documentation for personal or premeditated gain.

7.1.3 Cyber Kill Chains®

A Cyber Kill Chain®is based on a traditional Kill Chain used by the United States Military which implements a framework that allows them to: [1] indentify a target or attacker upon which to act or retaliate, [2] a method of deployment in the attack or retaliation, [3] the required strength and force to attack or retaliate with (based on the equipment and structural integrity of the opponents (attacker)), and [4] deployment and execution (** LITERALLY ** USA!, USA!!). Breaking this chain is regarded 'breaking an opponent's kill chain'. This, almost word-for-word can be applied to the soyber environment.

Often, the cyber aspect of this concept refers to the Advanced Persistent Threat (APT) concept where there may be stealth intrusion in a system where attacker may go undetected for a long period of time. The Cyber Kill Chain®considers this in great detail and aims to implementat relevant factors of the APT response life cycle in order to avoid detection. The chain takes form as follows: *reconnaissance*, *weaponization*, *delivery*, *exploitation*, *installation*, *command and control*, *actions upon objective*; much like the militant version.

Reconnaissance takes into account the current state of the target. That is, are they active or passive? Do they have their shields up or shields down? Will they be able to respond fast, respond at all? This is where the attacker in effect identifies the vulnerabilities of the target by gathering as much relevant information on the methods of entry as possible. For example, this may include gaining user entry details or searching for a 'backdoor', as seen in malware and malicious attacks.

In the weaponization stage, the attacker identifies which methods best take advantage of the openings and vulnerabilities identified in the reconnaissance segment. As expected, weapons include malware such as viruses, worms, backdoor trojan entry, DoS attacks, etc. These will be specific to the vulnerabilities on a what-fits-where-basis.

Delivery refers to when the attack weapons are utilized. Just as is the case in the military, the chosen weapons must be transported in a suitable vessel. Like you wouldn't send 50 armed men to battle in a 2012 BMW 535d station wagon, you wouldn't send a phishing note on a physical flash drive. The vessels would be too weak. Relevant vessels such as e-mail, phsical drives, direct human intrusion, etc., must be selected appropriately.

The exploitation portion of the process refers to the time at which the weapons are triggered. For example, a time-bomb or logic-bomb goes off, or an e-mail is opened which downloads malicious software, etc. Exploitation is considered instantaneously successful if the system, machine and/or target network suffers negative effect due to the weapons being active.

The installation period occurs when there is a virtual 'backdoor' open for the attacker to exploit. That is, when a virtual port is opened which allows the attacker remote access to the system or network target without having to repeat another exploitation, and allows them to transmit further malware or gain data from within.

It's at this point where the attacker can take command and control in the sense that the malware they have/are transmitting allows them to either automatically or manually manipulate and control the purpose or the function of the targets system or network. From here, the attacker completes the seventh and final step of taking action upon the objective. This is where they achieve the goal they set out to complete by either exfiltrating data, destroying data, controlling data, or encrypting data for ransom for personal gain.

Defensive action is often tagen by potential targets against this kill chain system. Its form is as follows: [1] they seek to *detect* when there is an attacker present in their system or network; [2] *deny* permissions and access to these attackers to the best of their ability' [3] *disrupt* incoming or outgoing transmissions of data which they did not authorize and are likely to have been inflicted by the attacker; [4] *degrade* the attacker by deploying a counter-attack by perhaps creating misleading 'backdoors' for attacker to be tricked by; [5] *deceive* the attacker by implementing the previous step and regain control of the system or network; and [6] *contain* the issue by protecting data and quickly analysing and repairing or regaining anything lost or maliciously recieved. A successful implementation of this protocol will in effect 'break the chain' of the opponent.