**Exam 2**

Secure email (example of how we combine symmetric and asymmetric encryption to send an encrypted message). Understand the theory of how the two common systems we studied work and know the differences between them

PGP

Generate your own keys

You are responsible to distribute

○ In person or through key servers

○ Key signing parties! 

S/MIME

Generate your own keys

Have your public key signed by a CA

Start sending signed email

○ Your public key is sent along with a signed message

Know diagrams

---------------------------------------------------------------------------------------------------------------------------------------------------

TLS Handshake

* know protocol control flow (given a diagram, reason from it)
* know record format
* know what data is exchanged at what phases and it’s composition.
* be able to explain the parts of a cipher name e.g.

Auth [ Alg., Strength, Mode]

TLS\_ECDHE\_RSA\_WITH\_AES\_128\_GCM\_SHA256

Key exchange Cipher MAC (SHA) or PRF (SHA256)

RSA

› Client generates pre-master secret & sends it to server encrypted with servers public key

› Client doesn’t have proof it’s valid till they get the message.



DHE

› DH shared key is the pre-master secret

> Pre-master secret and random values used to compute master secret

> Provides perfect forward security.

Pre-master secret and random values used to compute master secret 

Master secret and random values used to compute key block material › Key block contains 4 or 6 keys;

Two keys for AES, 2 keys for MAC, 2 keys (values) for block cipher mode if needed

-Review Questions

How many shared keys are derived between a client and a server that establish a TLS session?

› Each side generates 4-6 keys 

How does the server prove ownership of its private key?

› Implicitly by decrypting the pre-master secret (given from client) and finishing handshake 

How does the client prove ownership of its private key when client authentication is (rarely) used?

› Send digital signature to the server 

What is the pre-master secret?

› Who creates it? Client

› How is it securely transmitted? 

What is session resumption?

› How does it differ from a regular SSL handshake? 

When do the client and server start encrypting traffic using symmetric encryption?

› After they have finished the message/ whole exchange.

See the picture of the 4 phases

---------------------------------------------------------------------------------------------------------------------------------------------------

Buffer Overflow Attacks

* Stack Smashing Attack

During a function call, the return address is pushed on the stack

An attacker overflows a buffer (local variable)

The return address on the stack is overwritten to point to an existing function or to injected code

During the function return the instruction pointer is set to the new return address value stored on the stack, not the original return address that was pushed on the stack as the function was called

* Canaries

A variable in the code that will be the first to change should there be a buffer overflow or some other attack. The compiler checks this value before executing. Terminator, Random, and XOR canaries are random values created with addresses in the code, so that an attacker can’t just jump over the canary.

* What is a NOP sled? Why/How is it used?

It is put into the code an attacker is manipulating/ put into the command line so as to have the stack pointer be directed to a certain address. This makes attacks easier to do , since the attacker doesn’t need to point to the exact correct address, just close to it.

* Defenses (Prevention, and Mitigation, and what the difference between these is)
* Understand x86 stack layout calling conventions as seen in assignments

See the stack assignment

* Shown some debugger output, be able to reason about the function call

diagram and the state that the program is in and will be in

Binary extraction attacks and the use of a debugger.

---------------------------------------------------------------------------------------------------------------------------------------------------

Passwords

* Online vs. Offline attacks

Online is susceptible to browser/server shutouts, but don’t need to worry about salts.

Offline attacks can be computed faster, but must also compute salts.

* How to calculate the cost of an off-line attack

(Number of characters available ^ length of password) \* Salts

* Unix password files - how do they work?

Crypt. Creation and Login functionality. L

https://wiki.cs.byu.edu/media/cs-465/passwords.pdf

* Lamport's Hash - how does it work? How to attack it?

<http://lodestone.org/people/hoss/ops/node5.html>

One time password scheme.

Attack on Lamport’s Hash 

Small n attack

Active attacker intercepts servers reply message with n and changes it to a smaller value Attacker can easily manipulate the response (repeatedly) to impersonate Alice 

Eavesdropper captures Alice’s hashed reply and conducts off-line attack. Dictionary attack. 

Replay Alice’s response to other servers where Alice may use the same password

Thwart using salt at the server – server hashes pw || salt and sends n and the salt to Alice during login

Salt also permits automatic password refresh when n reaches 1

* Salts - what and why?

Why do Unix password files use a salt?

Prevents the identification of identical passwords

○ Provided each user has a different salt

All password guesses are salt-specific

○ Guess made with one salt aren’t helpful for another

○ Increases the cost of offline attack to crack any password in the file

○ Increases the size requirement for a precomputed database of hashed passwords

Password Attacks with Salt 

How many guesses do password attacks need when a salt is used?

* Off-line attack – one attempt for each unique salt in the file 

How does the salt impact on-line attacks? It doesn’t 

How does the salt impact an attempt to crack a specific user’s password in the file?

* It doesn’t change the number of attempts, but it does increase the size of a pre-computed database of passwords or rainbow table

Attacks against, and countermeasures for programs that hold passwords

-Brute-force  -Dictionary - Substitution: password, passw0rd

Ken Thompson compiler hack - how it works

1. Describe briefly and clearly how the attack works

The goal of the attack is to produce 2 “Trojan horse’ self-reproducing programs into someone’s C compiler. To do this, it makes the compiler accept these Trojan horses by changing them into legal C binary code when the system recompiles. That way, the login command will remain bugger with no trace of change in source anywhere.

2) If you suspect that your machine has been compromised, what should you do about it? Ken Thompson suggests that when considering such an attack as he has described, you should check for malicious code placed in your compiler. In general, one should use reliable backups for your information, change your passwords, install legitimate anti-virus protection, re-install your OS, etc.

3) What other kinds of software like compilers do we usually trust that have the potential to be compromised?

Other such software can include (but is not limited to) most any program-handling program like ones assembler, loader, or even ones hardware microcode

Principle of Least Privilege and how it relates to Defense in Depth

You allow your code to run only a few processes and have very few privileges. If you are attacked, the attacker has a very limited access to your resources.

---------------------------------------------------------------------------------------------------------------------------------------------------

ANYTHING in any of your projects, or homework assignments.

Review Exam 1 problems

* MAC/HMAC, hashes and how they are implemented, and how they are not the same