Web Application Security: The Devil is in the Details

Session I of III

JD Nir, Security Analyst

independent security evaluators

Why is this important?

Agenda

- About ISE
- Web Applications and Security
- What is OWASP?
- Vulnerabilities
  - Injection-based attacks
  - Broken Authentication and Session Management
  - Cross-Site Scripting (XSS)
- Q&A

About ISE

Perspective

- White box

Analysts

- Hackers; Cryptographers
- Reverse Engineers, etc

Research

- Routers; NAS; Healthcare

M&E Customers

- Content Owners; Vendors; Supply Chain

Talks

black hat

RSA CONFERENCE

DEFCON

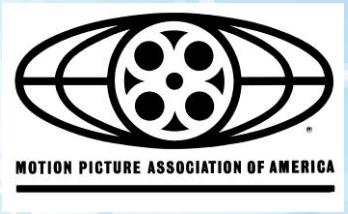
DERBYCON

BSIDES WASHINGTON D.C.

ISE in M&E

Talks

Involvement



Web Application Security

- What is a vulnerability?
- What causes vulnerabilities?
- Understanding a vulnerability:
  - What is it?
  - Why does it matter?
  - How do I detect it?
  - How do I prevent it?
  - What nuances should I understand?

What is OWASP?

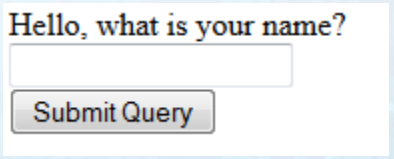
OWASP

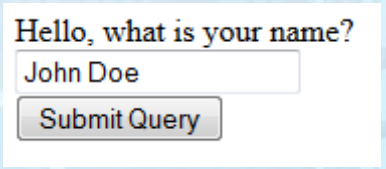
The Open Web Application Security Project

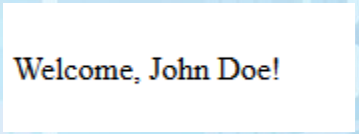
- Not-for-profit, free, open source resources
- The OWASP Top Ten

INJECTION

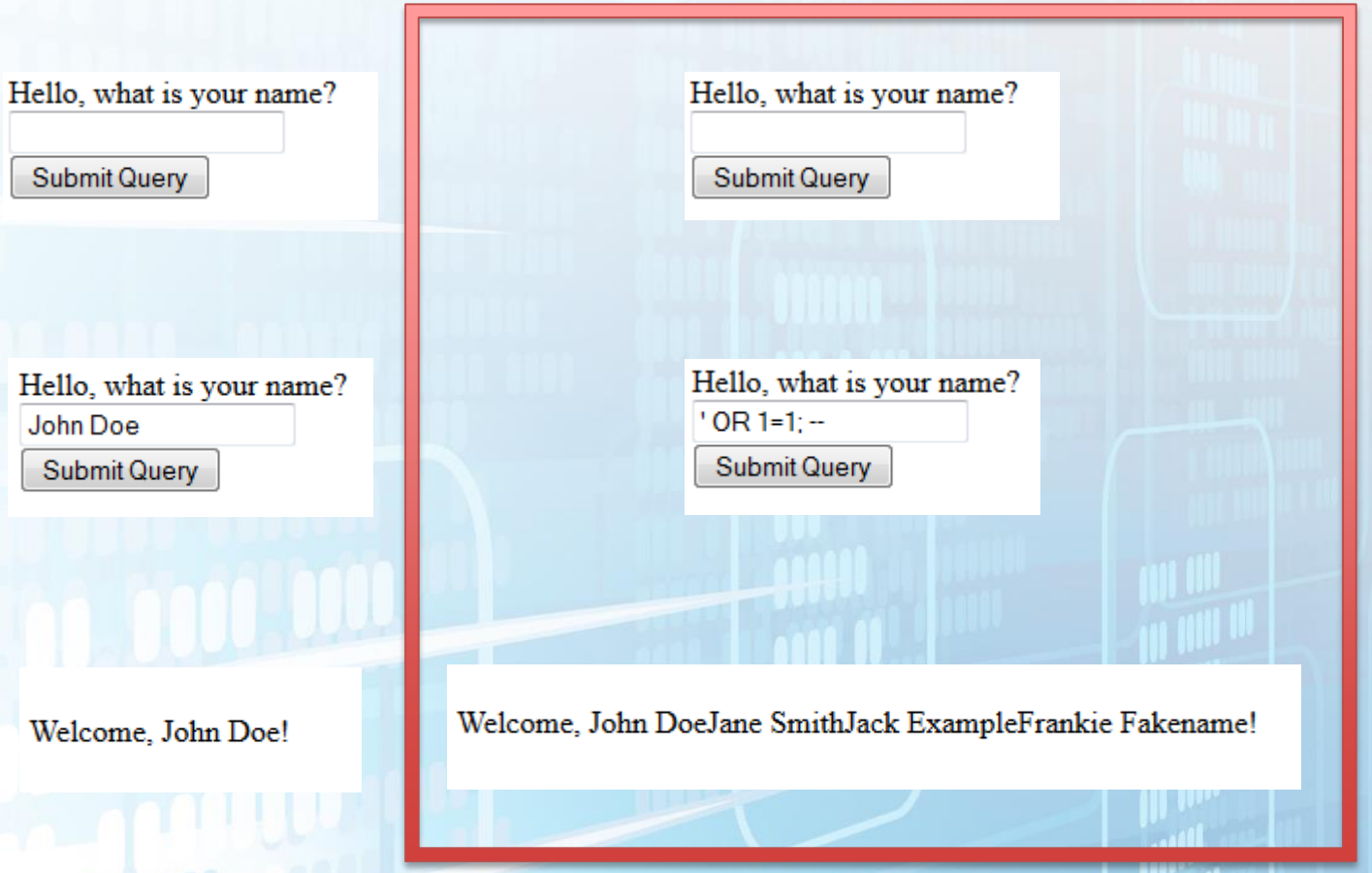
Injection – What is it?



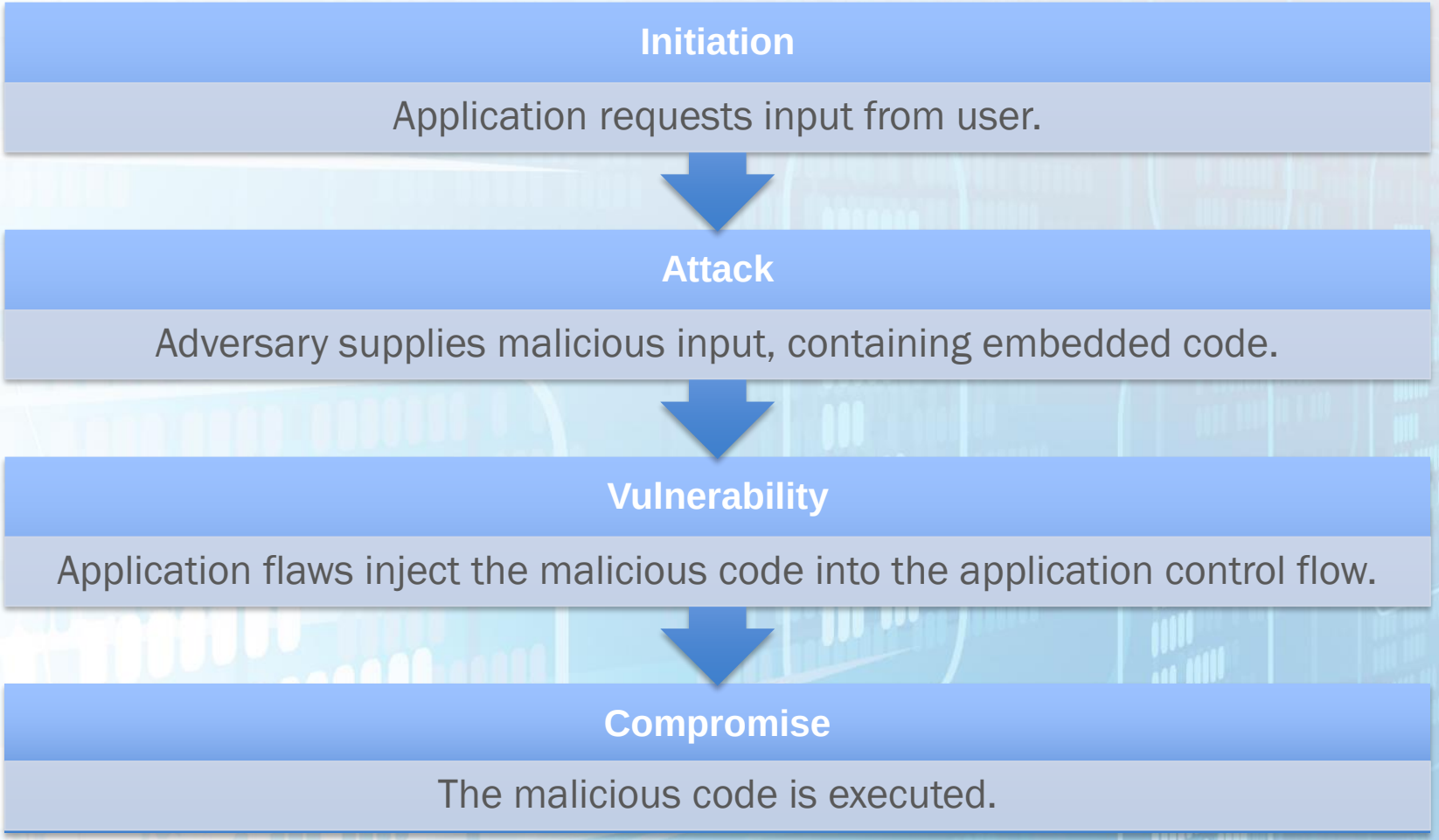




Injection – What is it?



Injection – What is it?



Injection – Why does it matter?

Asset Theft	Asset Destruction	Machine Manipulation
• Passwords, credentials • billing information	• Data	• Gain control of the entire machine.

Injection – How to detect it

Automated

- Free detection tools are available.
- Best known for SQL inj. is `sqlmap`

Manual

- Manual code reviews are the only way to discover subtle errors that may allow for custom attacks.

Injection – How to prevent it

Method #1: Sanitization

- Input is “cleaned” before getting processed.
  - “Validation” restricts users to a set of “safe” inputs. Some legitimate users may encounter errors.

    e.g. Bob O’Neil is told his name can’t contain ’
  - “Escaping” replaces potentially dangerous characters with codes. Before use, the data must be “unescaped.” e.g. O’Neil → O\’Niell

What is a “safe” set of characters?

Injection – How to prevent it

Method #2: Parameterization

- Use “fixed” commands with variables holding the place of user data. The attacker cannot inject code because the command cannot be changed or have additional code added to the end.

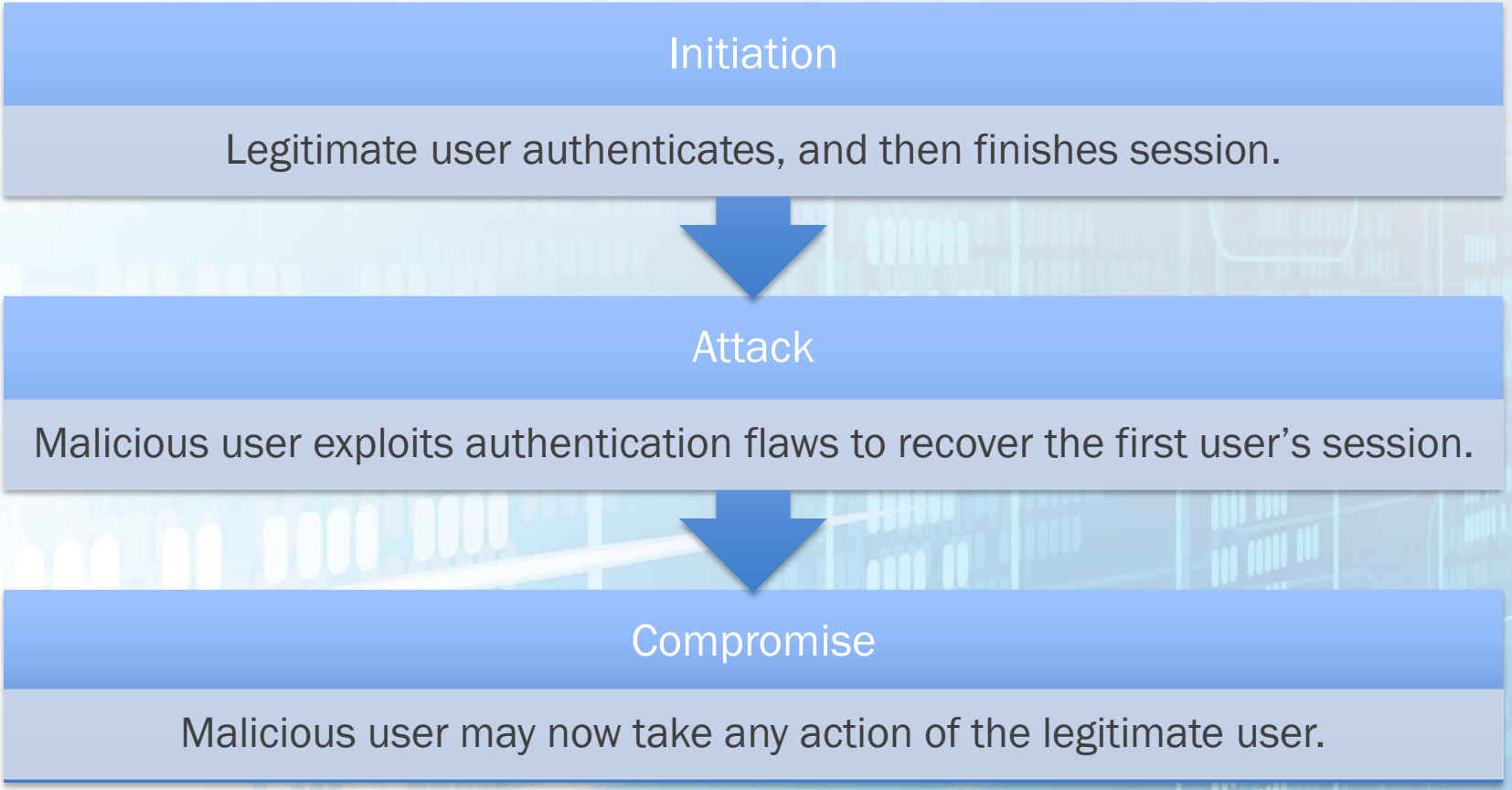
When available, parameterization is the safer method of preventing injection.

Injection - Nuances

- Suppressing error pages is not enough. Blind injection attacks can extract data even without output!
- Changing backend databases changes the set of “safe” characters. Make sure to also switch to the proper sanitization functions.
- Injection goes beyond SQL/LDAP. Command injection is far more dangerous.

BROKEN AUTHENTICATION AND SESSION MANAGEMENT

Broken authentication – What is it?



Broken authentication – What is it?

Some examples:

- Session IDs included in URL, saved in browser history
- Session IDs don't time out
- Passwords are sent over an unencrypted connection

Broken authentication – Why does it matter?

- Sessions are double-edged sword.
- Users **trust** applications.
- Common attack target.

Broken authentication – How to detect it

Automated

- Very few tests available.

Manual

- Most effective method.
- Important to stay up-to-date; new technologies and features often open new attack avenues.
- Proper logging can alert to a breach – if logs are monitored.

Broken authentication – How to prevent it

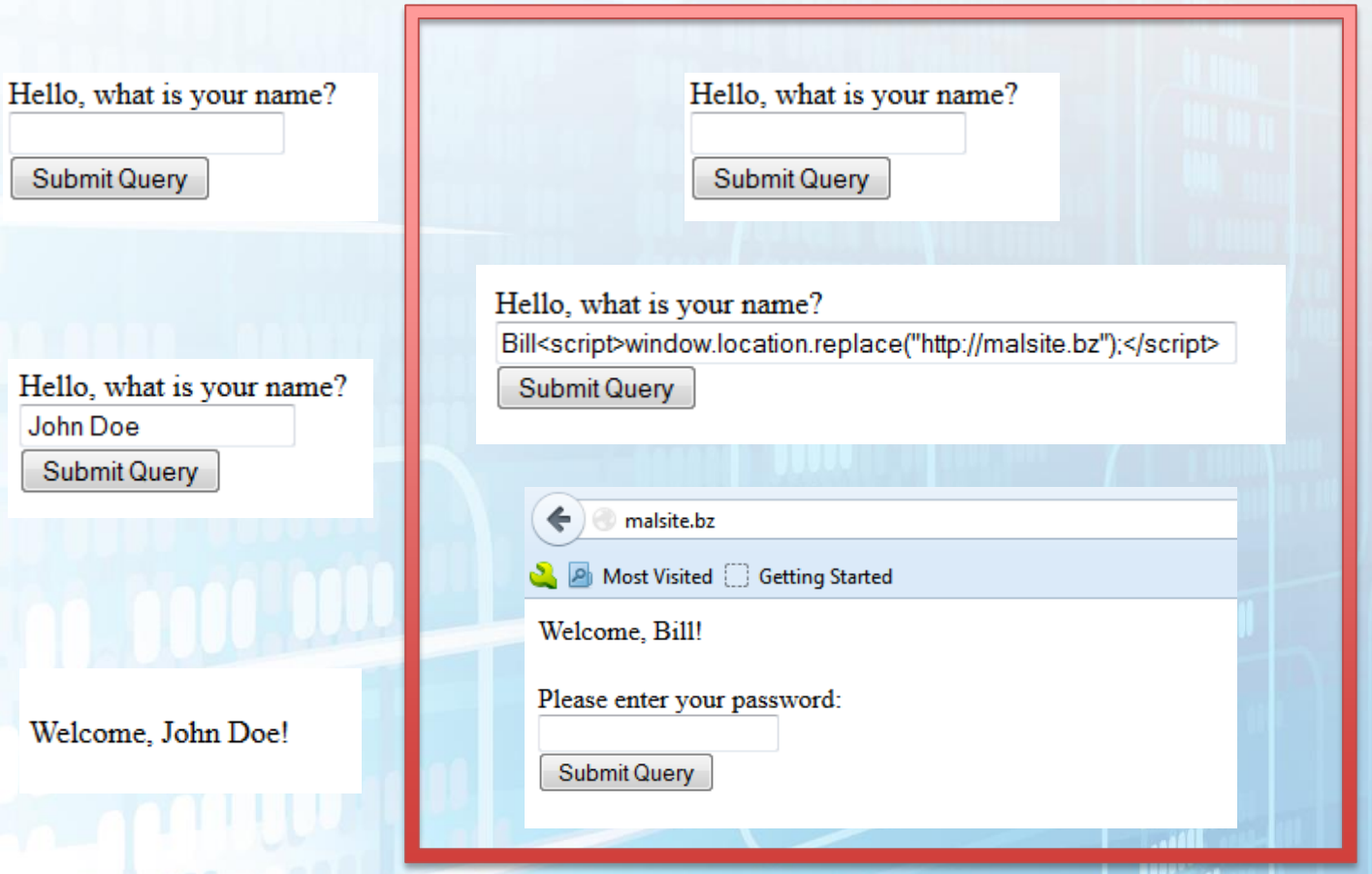
- Do not store plaintext passwords.
- Keep session IDs secret, random, and short-lived.
- Regular security evaluations, either in-house or contracted, by security professionals who are up-to-date on latest attacks.

Broken authentication - Nuances

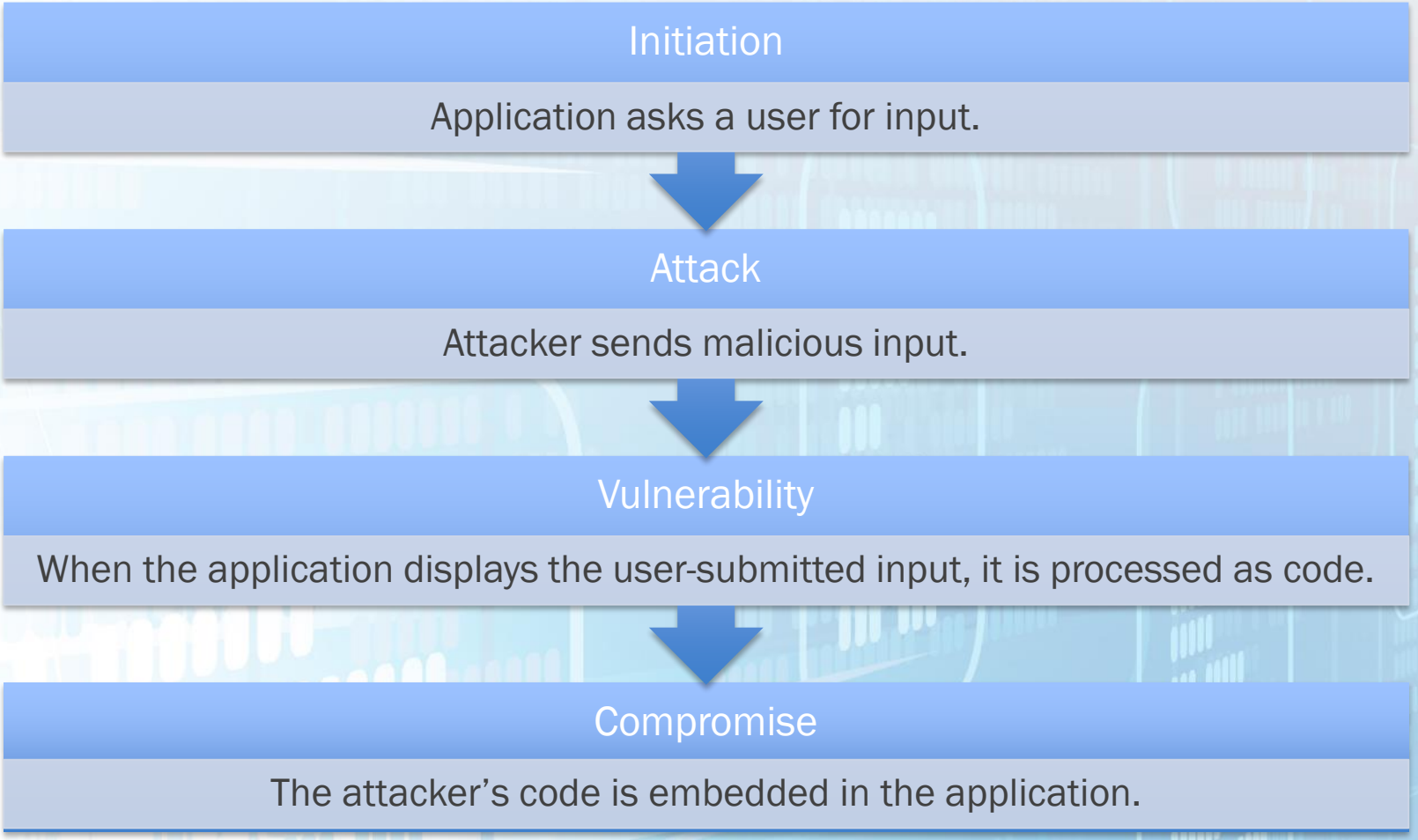
- Client-side authentication alone is never secure.
- Beware of development backdoors.
- “Defense in Depth.”
- Session and Authentication vulnerabilities vary drastically in risk. Understanding how attacks work is crucial when deciding how and whether to address them.

CROSS-SITE SCRIPTING (XSS)

XSS – What is it?

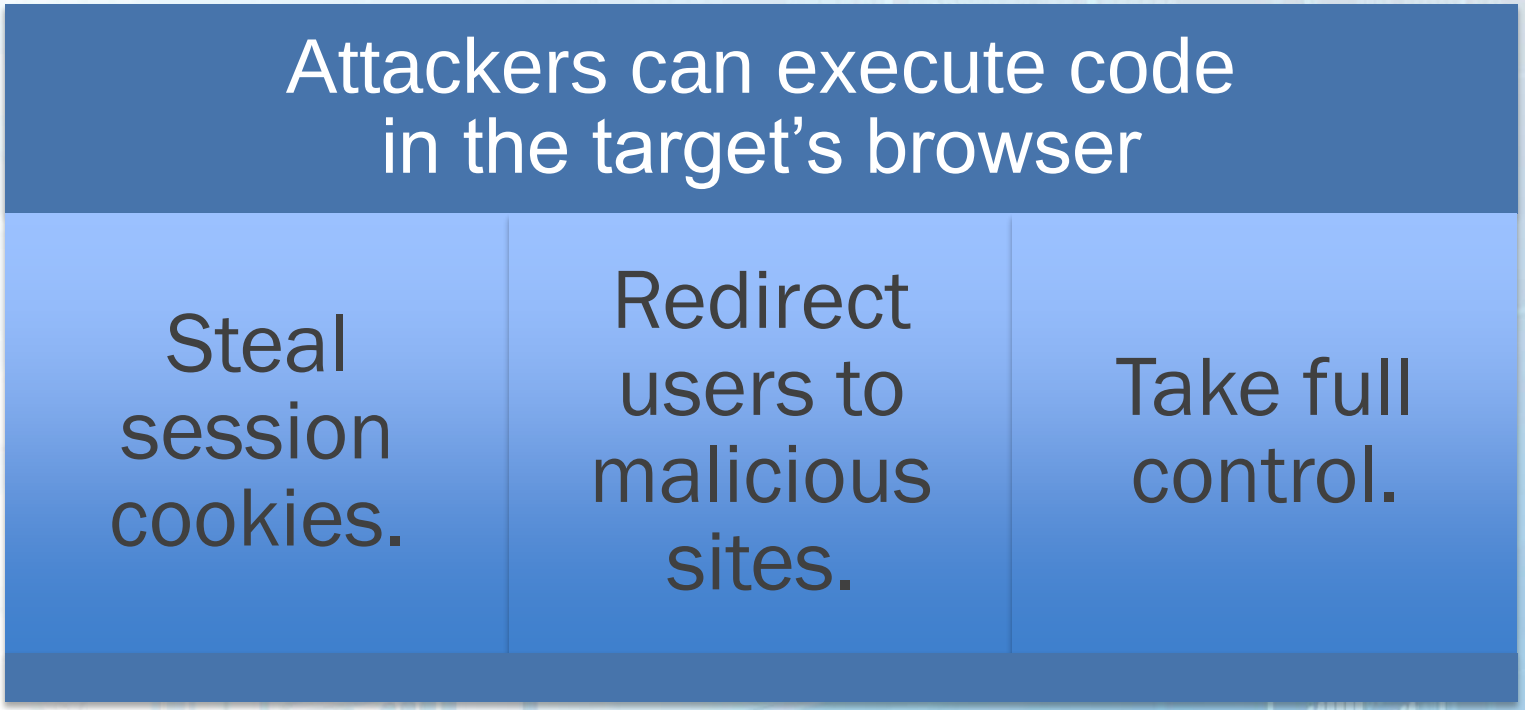


XSS – What is it?



XSS – Why does it matter?

Legitimate users trust your application. Attackers prey on that trust.



XSS – How to detect it

Automated

- Some automated tools exist, but most only catch easily-exploited vulnerabilities.

Manual

- As always, manual reviews are the only way to detect sophisticated attacks.

Cross-Site Scripting is an example of Injection. Similar methods apply.

XSS – How to prevent it

- Usually XSS vulnerabilities cannot be addressed with parameterization. This means filtering must be used.
- When possible, use validation.
- Look for heavily used escaping libraries.
- “Defense in Depth”

XSS - Nuances

XSS type #1: Reflected attacks

- Best known.
- An application reflects user input onto a page. An attacker tricks a target in to navigating to a page with malicious input.

XSS - Nuances

XSS type #2: DOM based attacks

- Similar to reflected attacks, but more dangerous.
- Injection occurs on client's machine instead of the server.
- Attacker can act as user's browser, including making changes on user's computer.

XSS - Nuances

XSS type #3: Persistent attacks

- The most dangerous.
- When an application saves user input and displays it to other users (e.g. a message board), many more potential victims are exposed to an XSS exploit.

XSS - Nuances

- Understanding data flow is crucial.
  - Base 64 or other encodings dodge filters
  - Data must be properly escaped and unescaped.
- Client-side protections can be evaded and are not sufficient. However, they are still necessary to prevent DOM based attacks.

contact@securityevaluators.com

Questions?

Next Session:

Insecure Direct Object References

Security Misconfiguration

Sensitive Data Exposure