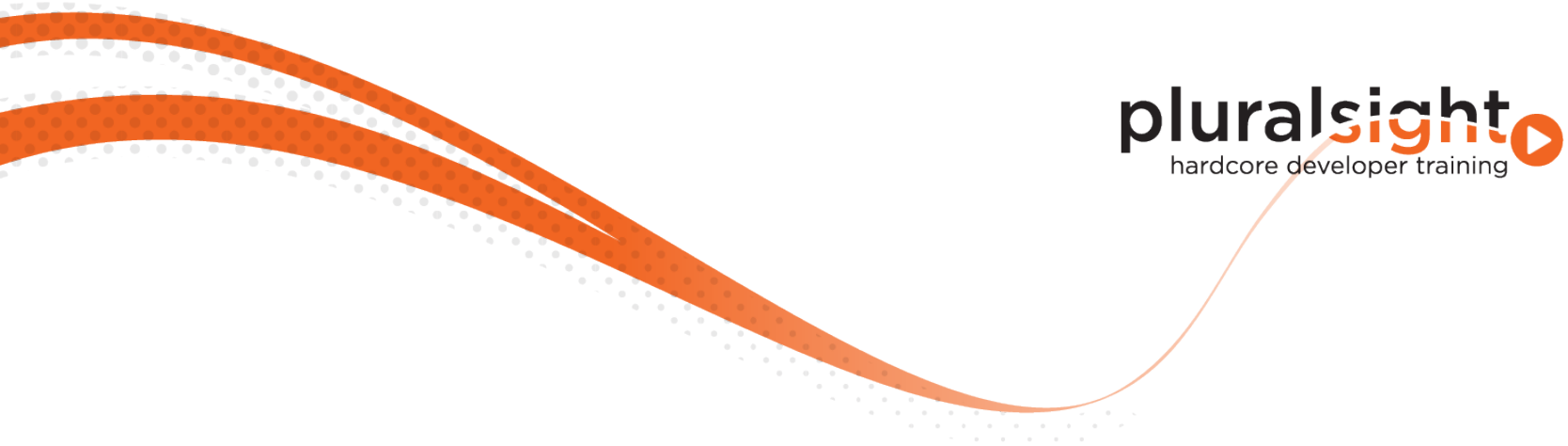
Insecure Direct Object References

Troy Hunt
troyhunt.com



pluralsight
hardcore developer training

Outline

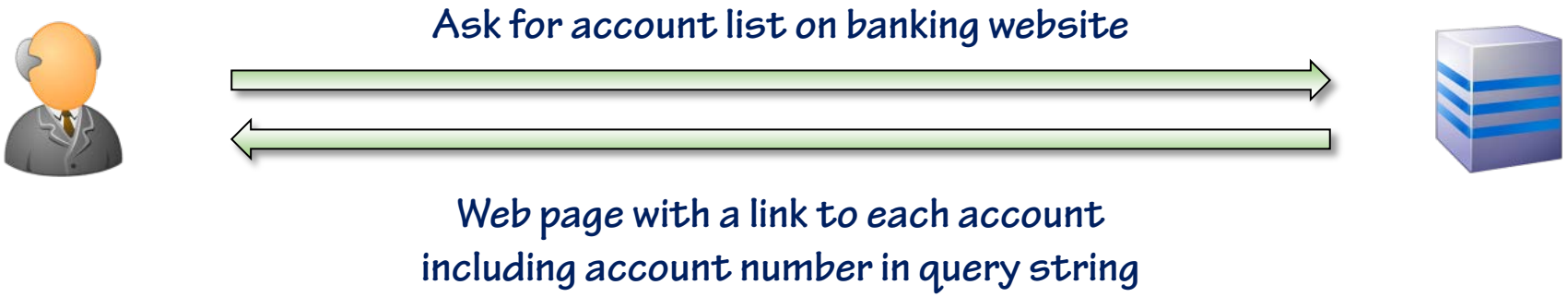
- How OWASP views the risk
- Performing an attack
- Understanding direct object references
- Implementing access controls
- Building an indirect reference map
- Obfuscation via surrogate keys

OWASP overview and risk rating

Threat Agents
—
Consider the types of users of your system. Do any users have only partial access to certain types of system data?

Understanding direct object references

- A direct object reference is an observable key used to identify an individual database record



- For example:
  - http://mybank.com/Account?id=534982345
  - http://mybank.com/Account?id=534982346
  - http://mybank.com/Account?id=534982347

The risk of direct object references

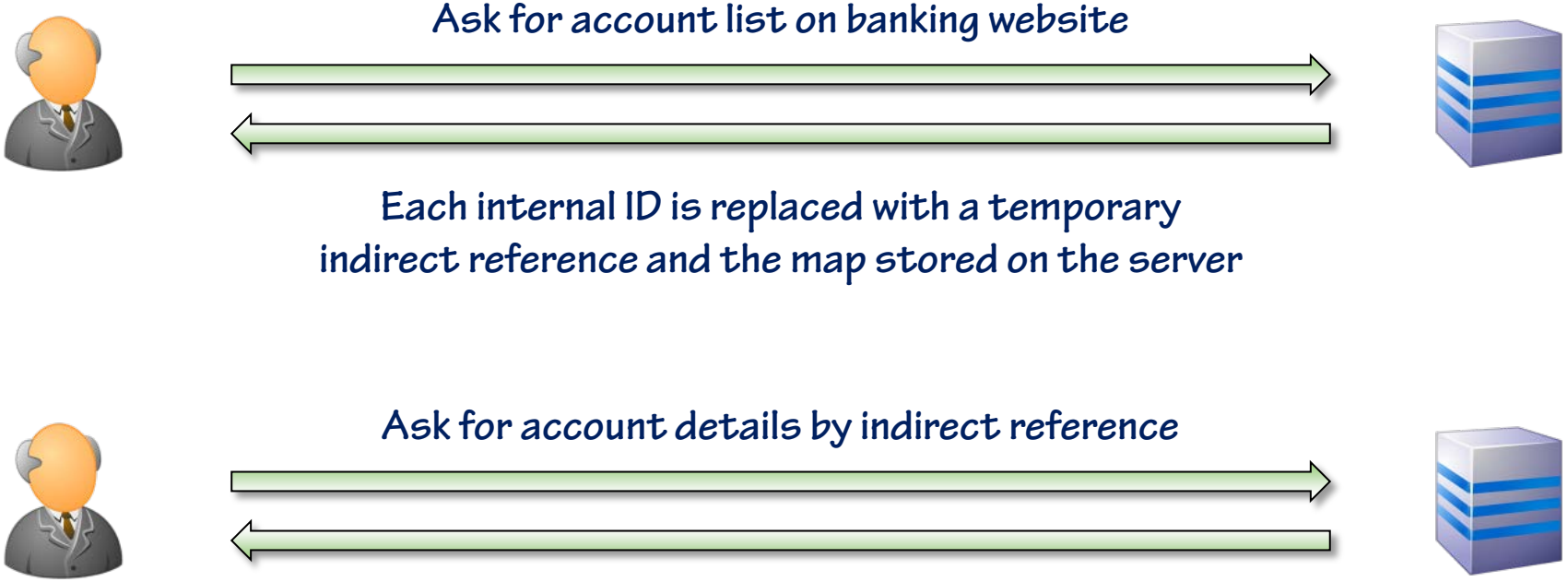
- **If a reference may be manipulated such that it refers to another record in an unauthorised fashion, a direct object reference risk is present**
- **Direct object references are usually:**
  - Patterns such as uniformly incrementing integers
  - Natural keys such as username
  - Discoverable data such as social security numbers
- **These may be simply guessed or enumerated either manually or by an automated script**

The importance of access controls

- Insecure direct object references are ultimately exploited due to lack of access controls
- Never assume a URL is “safe” just because it’s not immediately visible
- Particularly when IDs are enumerable
- Remember that URLs are also recorded at various stages of the request lifecycle

Understanding indirect reference maps

- An indirect reference provides an abstraction between what is publicly observable and the individual database record



Implementing the map

- **There are numerous constructs that can be used to store the map**
- **Consider factors that may influence the type of map storage used**
  - If there are multiple web front ends
  - Potential performance overhead
- **Important principles include:**
  - That the map is temporary
  - That the map is user specific
  - That the indirect reference is random
- **Examples of indirect references:**
  - http://mybank.com/Account?id=fyzFgYEzi2r97XRQojYbsXl78YKV1IsIHCHVWUnu2Lc1
  - http://mybank.com/Account?id=gZAH9ZD1JzVbaCsKJOkd68tHXhAeEQMVVP4xJ92pZkw1
  - http://mybank.com/Account?id=C0BXgoGw03oCiiREN5aJCgTtERBlxh7DlcBpwQRXjNk1

Obfuscation via undiscoverable surrogate keys

- **Integer and natural string types are vulnerable to enumeration**
- **A surrogate key that is not pattern-based can add further obfuscation**
  - A GUID is a good example
- **However, it *is* security through obscurity**
- **There are other issues to consider too:**
  - They usually don't perform as well on the database end
  - The storage requirement is higher
  - They still don't change the need for proper access controls

Summary

- **Insecure direct object references are ultimately about access control**
  - It always boils back down to insufficient authorisation
- **Indirect references can be used to conceal internal keys**
  - But they're *never* a substitute for access controls
- **Surrogate keys can assist in obfuscating IDs**
  - Bit it's still not an access control and has other downsides