Communication Networks (0368-3030) / Spring 2011

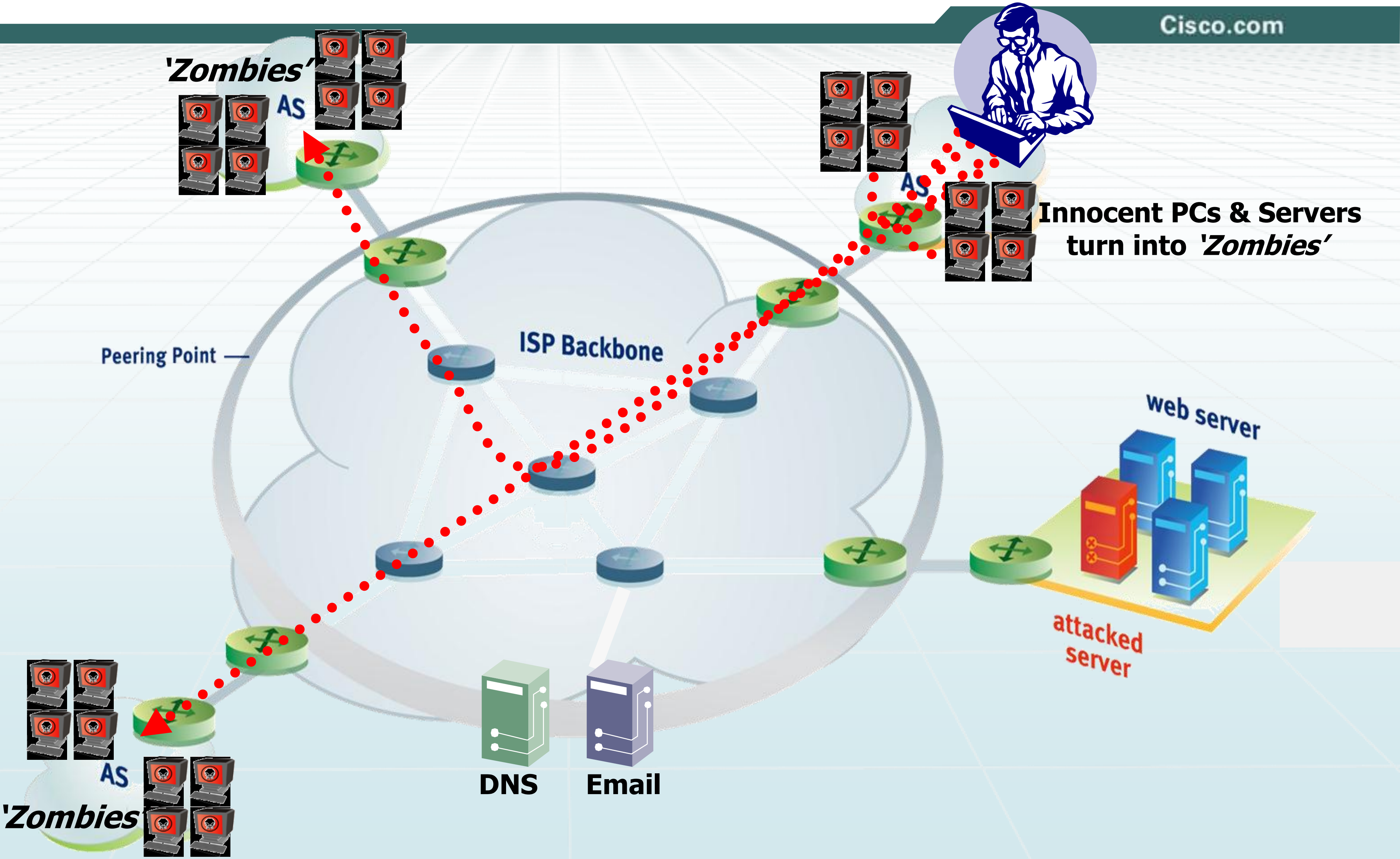
The Blavatnik School of Computer Science, Tel-Aviv University

Allon Wagner

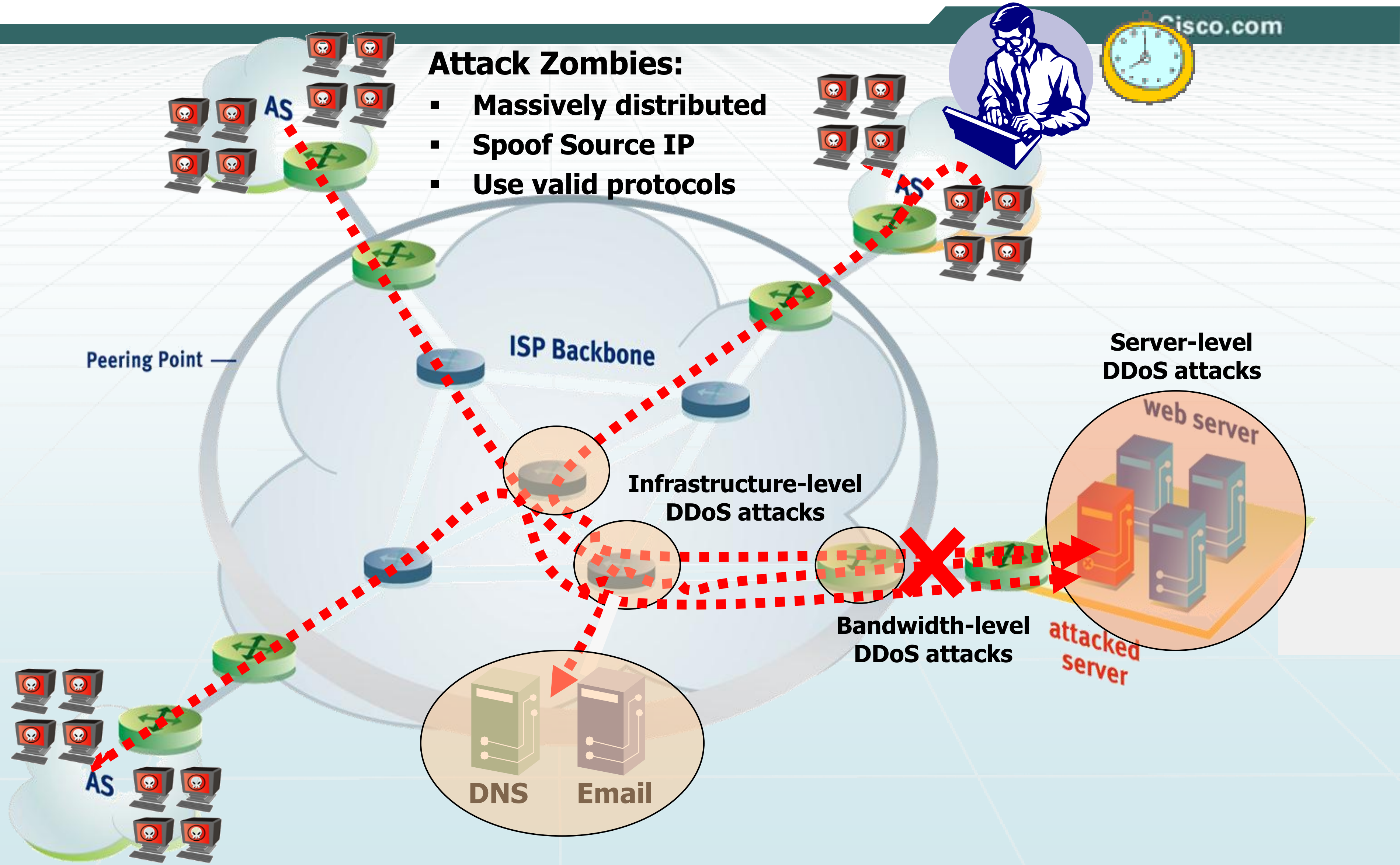
DDoS and Related Attacks

Several slides adapted from a presentation made by Dan Touitou on behalf of Cisco.

How do DDoS Attacks Start ?



The Effects of DDoS Attacks

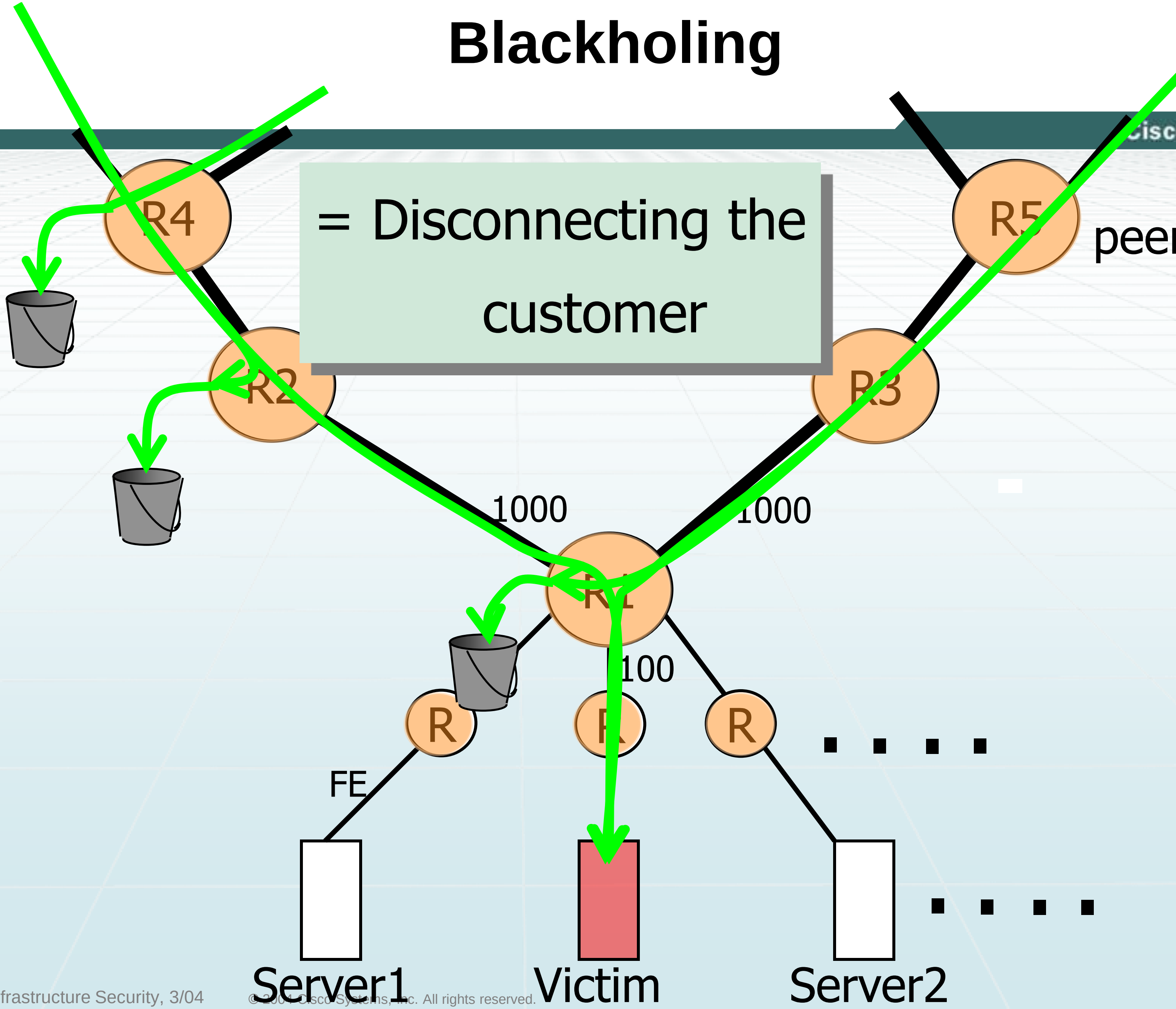


Motivation to attack

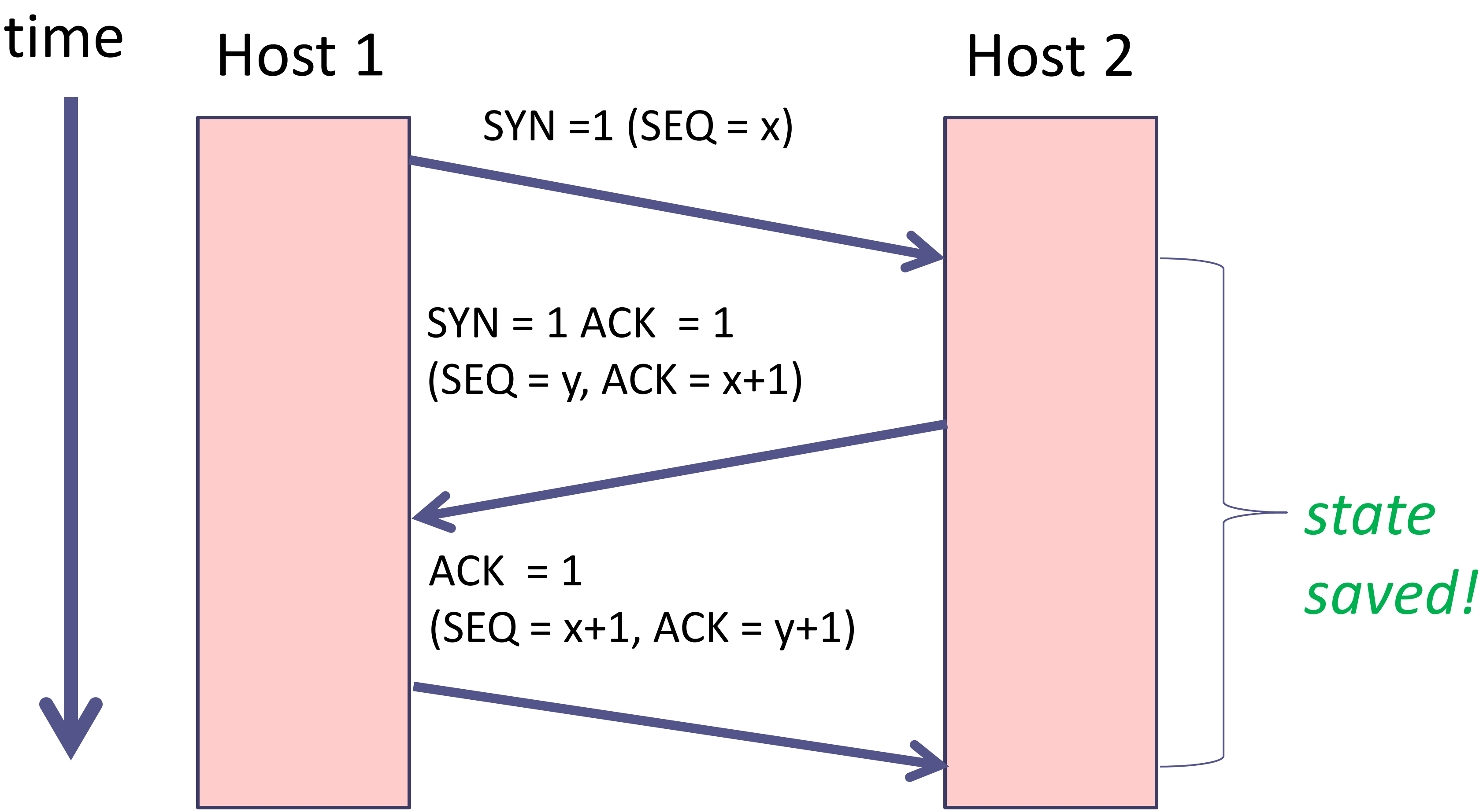
- Economically driven
  - Extortion
  - Zombie armies for hire
- Cyber-vandalism
- Cyber-terrorism / Cyber-war
- Backdrop for a more sophisticated attack
  - For example, an attacker brings a target down, and can then hijack its identity

Blackholing

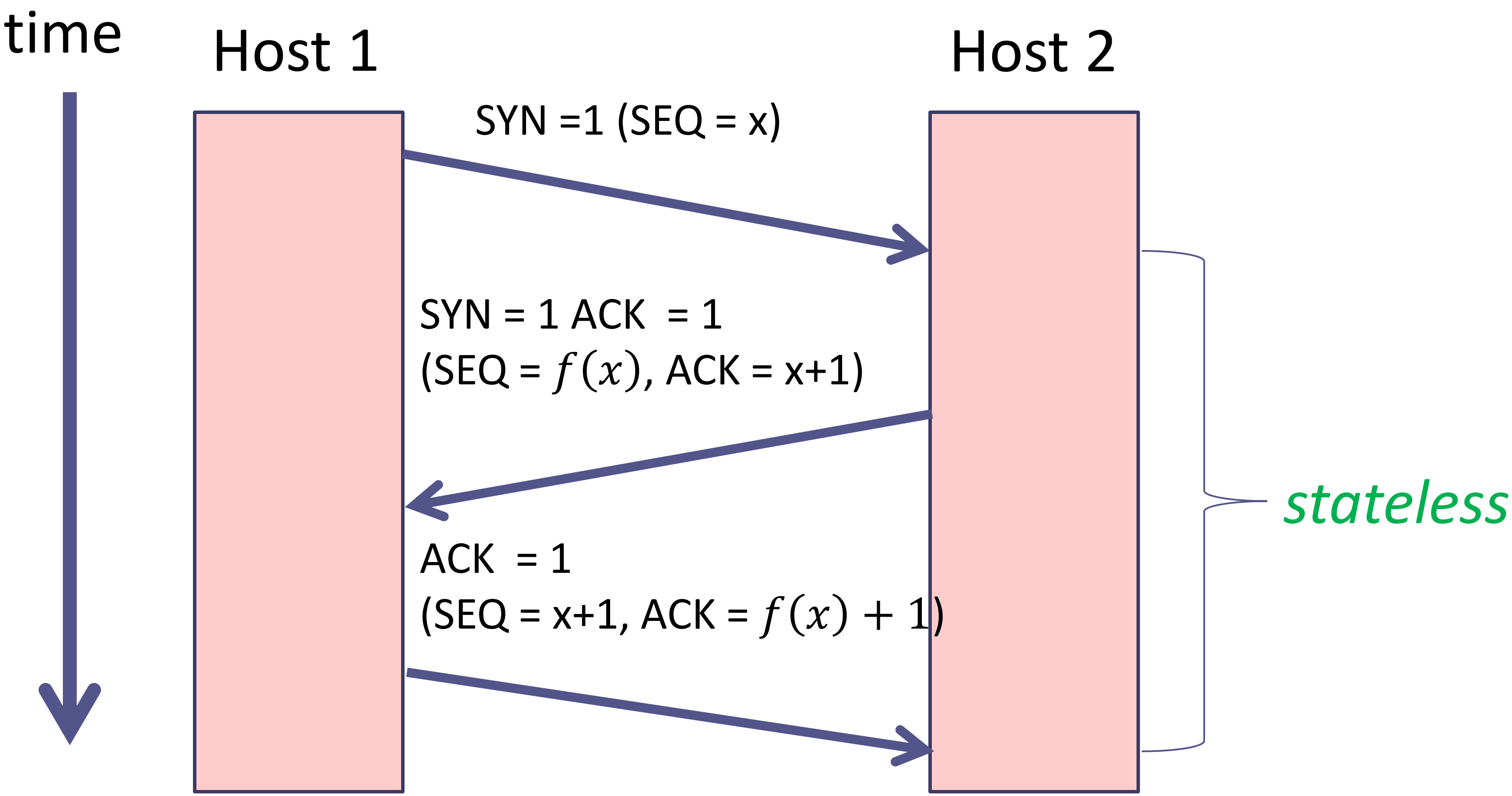
= Disconnecting the customer



Three-way handshake & SYN-Flood attacks



SYN Cookies – the idea



SYN Cookies (somewhat simplified)

- A client sends a SYN packet.
- The server does not choose a random SEQ for its reply. Instead, it calculates a $H(x)$ - a cryptographic hash of:
  - t – a slowly increasing time function (e.g increases every 64 seconds)
  - Server's IP and port
  - Client's IP and port
  - s - a secret
  - x – client's ISN
- The SEQ returned in the SYN+ACK packet is the concatenation $(t, H(x))$.

SYN Cookies (somewhat simplified)

- When a new client sends an ACK with ACK=y, the server decreases 1 and obtains:
    - t – allows it to ensure this is a recent request
    - the supposed hash result $H'(x)$
- It can recompute $H(x)$
- If $H(x) = H'(x)$ the client is legitimate and a TCP connection is opened

Exercise

- Why is t included in the cryptographic hash?
- To prevent replay attacks.
- Assume that Eve (an Evil attacker) wants to mount a DDoS attack against a server that does not include t in its hashes. Eve (and Eve's zombies) create millions of legitimate connections over a period of time, and collects $H(x)$ matching their data.
- When Eve wants to attack, she sends all these past requests simultaneously
  - ACKs imitating the 3rd step of the threeway-handshake along with their correct $H(x)$.
  - Plaintext field t simply says "now".
- The server cannot tell these are old requests.

Exercise (cont.)

- Why is t also given in plaintext?
- Because once a server gets the 3rd ack of the three-way handshake, it cannot know when the SYN-ACK reply was given to the client
  - i.e., what t was used to generate $H(x)$
- A malicious client still cannot forge $H(x)$ because it doesn't know s.

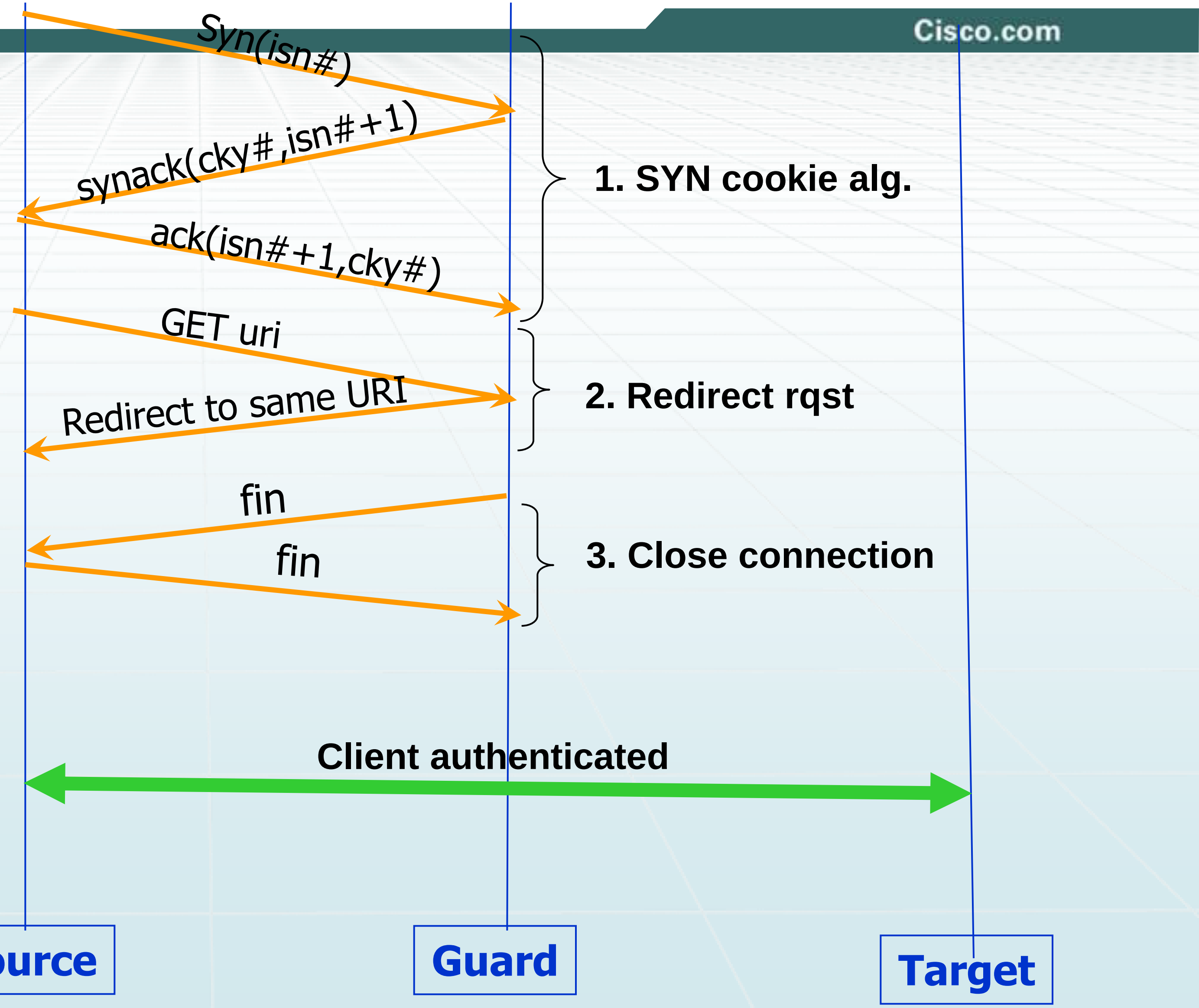
Anti-spoofing

- Spoofing – masquerading as a different network user
  - IP spoofing
  - DNS spoofing
  - ARP spoofing
  - …
- Malicious clients spoof IP addresses in order to mount DoS attacks.
- An idea to prevent (or at least hinder) spoofing: respond to the client in a way that forces it to reply.

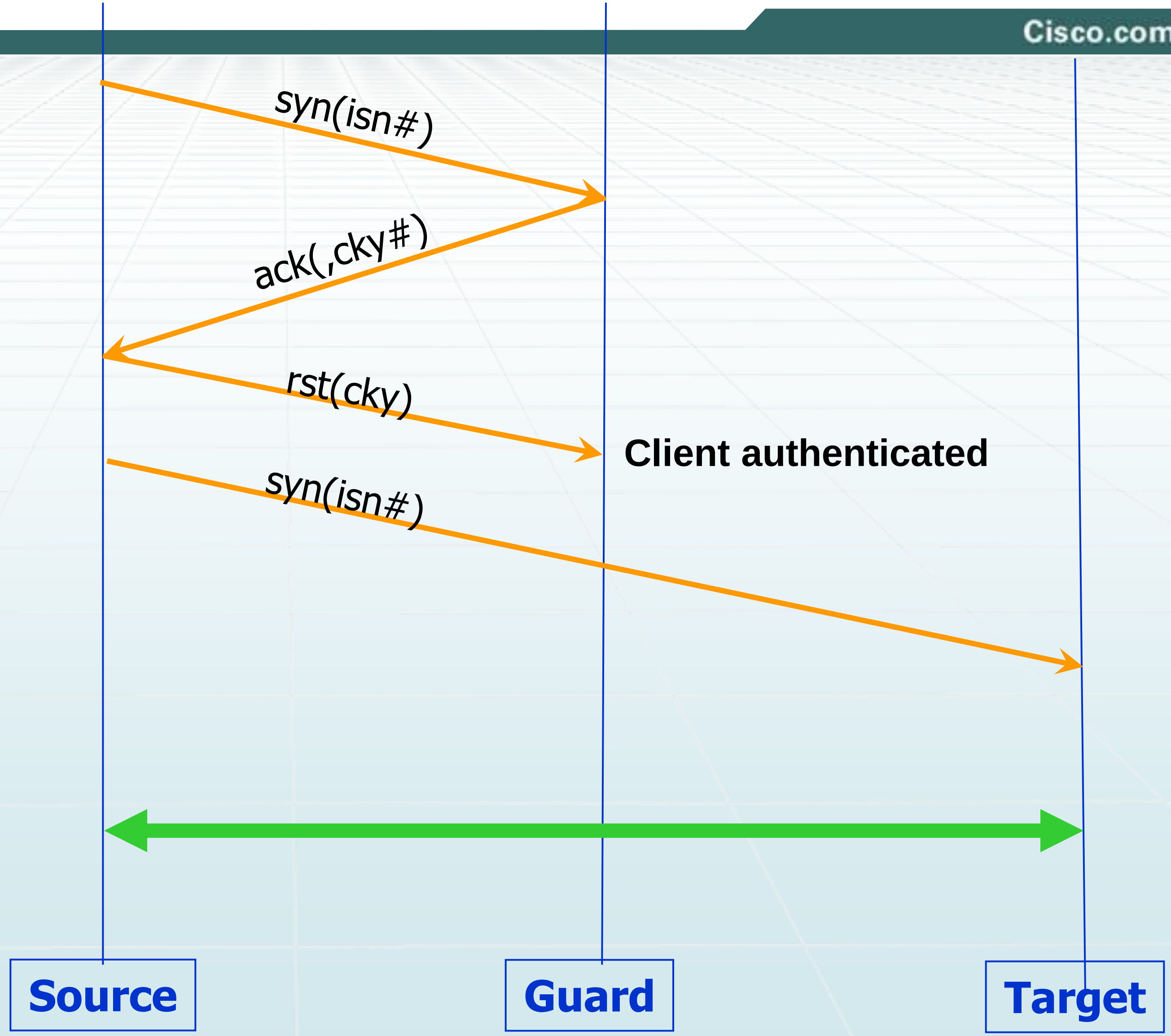
Anti-Spoofing Defense
- One example: HTTP

Antispoofing only when under attack

- Authenticate source on initial query
- Subsequent queries verified



RST cookies – how it works

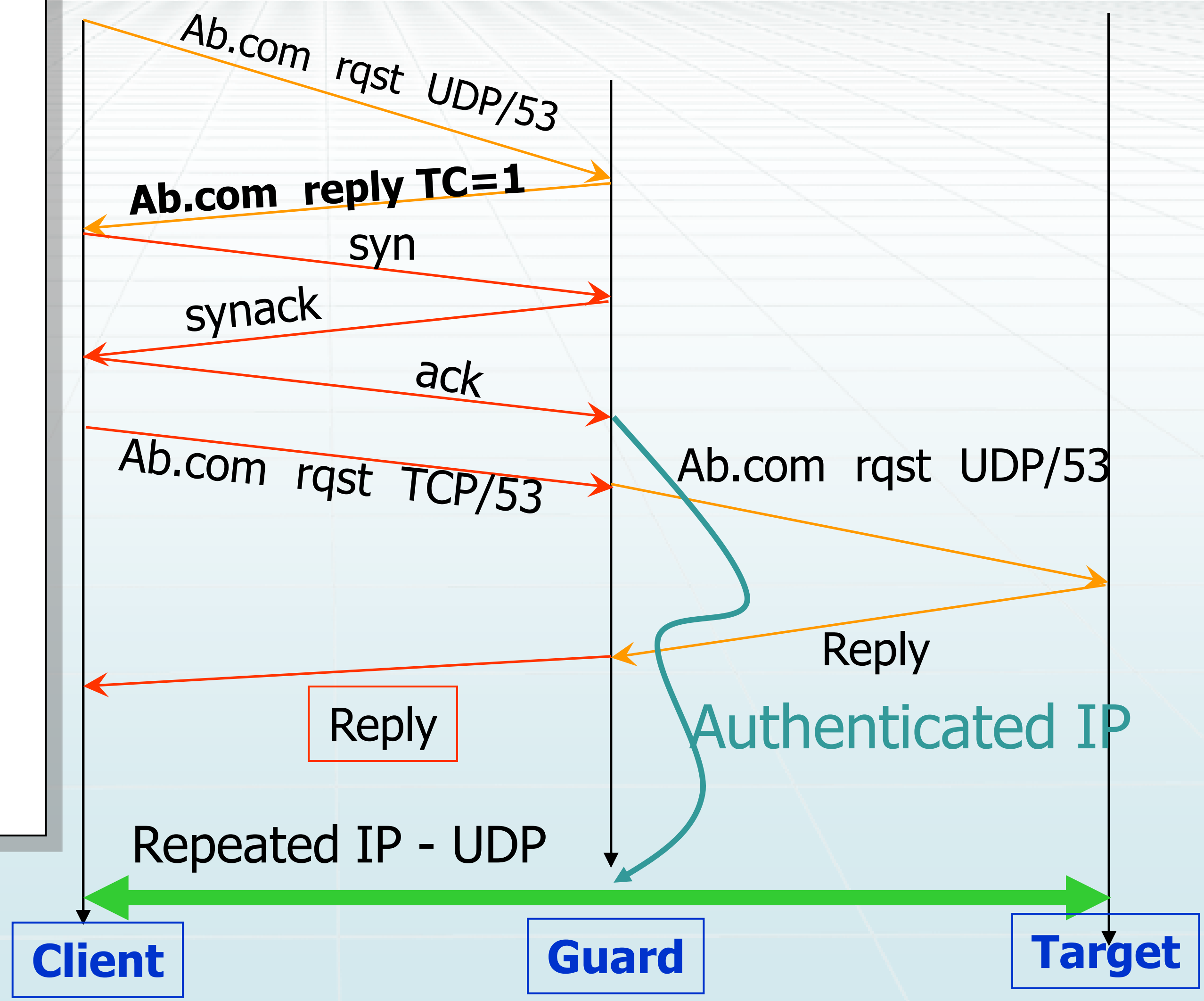


Anti-Spoofing Defense

- One example: DNS Client-Resolver (over UDP)

Antispoofing only when under attack

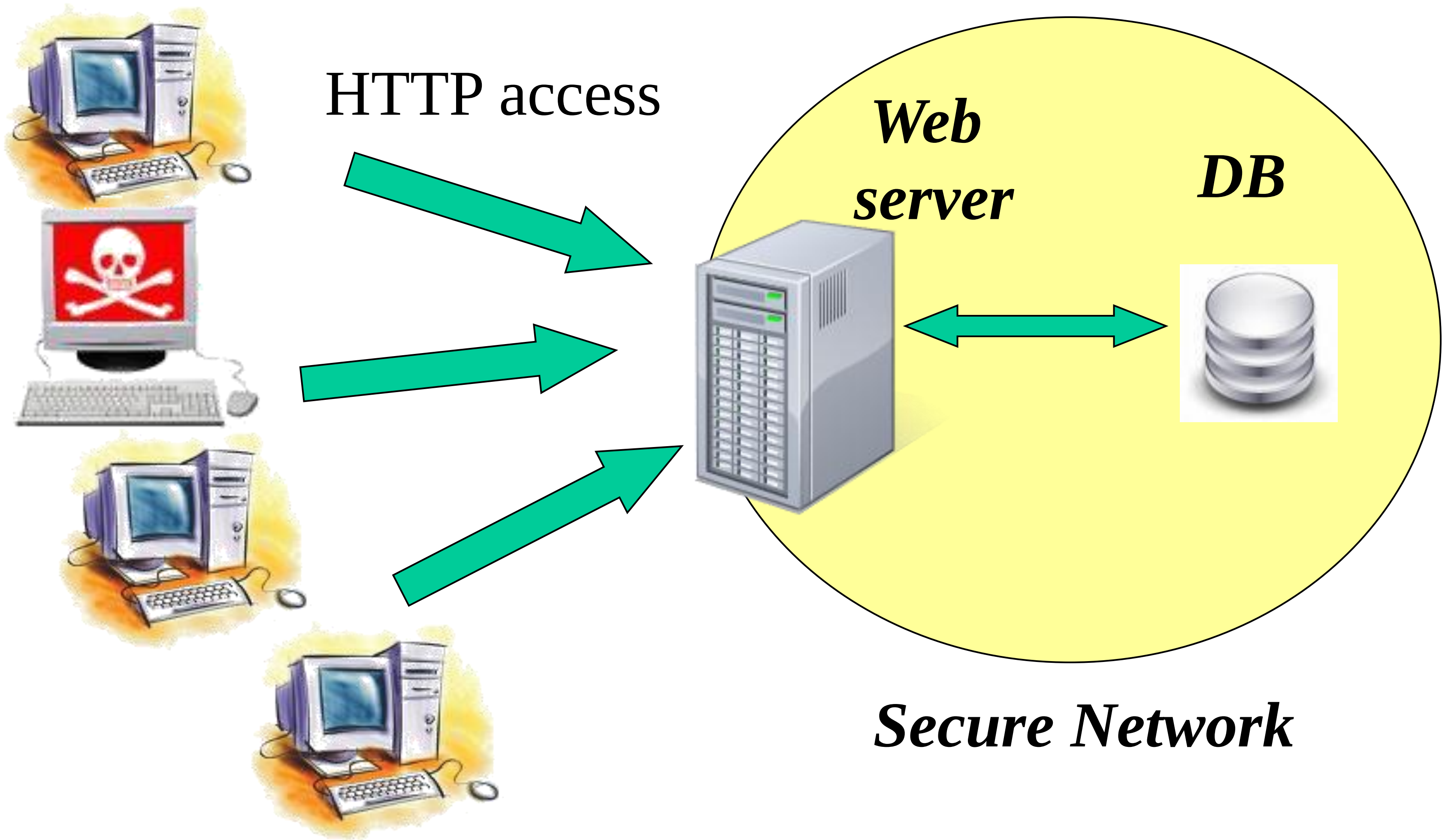
- Authenticate source on initial query
- Subsequent queries verified



Extra slides

SQL Injections - from an old talk I gave in the school

Our Objective – Prevent SQL Injection and XSS Attacks



SQL-Injection

- Benign:
  - `SELECT * FROM users WHERE name=`‘**alice**’ `AND password=`‘**1234**’
- Malicious:
  - `SELECT * FROM users WHERE name=`‘**alice**’ `AND password=`‘**1234’ OR ‘a’=‘a**’
- We got ourselves a list of usernames and their respective passwords, and can access the DB

SQL-Injection (cont.)

- Benign:
  - `SELECT phone FROM clients WHERE name='`**`alice`**`'`
- Malicious:
  - `SELECT phone FROM clients WHERE name='`**`alice'; UPDATE clients SET debt=0 WHERE name='eve';--`**`'`
- Information tampering. Can also be used for DB mutilation and information disclosure

SQL-Injection - Audit Evasion

- Benign:
  - `SELECT phone FROM clients WHERE name=‘`**`alice`**`’`
- Malicious:
  - `SELECT phone FROM clients WHERE name=‘`**`alice’; UPDATE clients SET debt=0 WHERE name=‘eve’;--`**`’`
- A skilled DBA will be able to track this!

SQL-Injection – Audit Evasion (cont.)

- Benign:
  - `SELECT phone FROM clients WHERE name=‘`**`alice`**`’`
- Malicious:
  - `SELECT phone FROM clients WHERE name=‘`**`alice’; UPDATE clients SET debt=0 WHERE name=‘eve’; --sp_password`**`’`
- MS SQL Server 2000 prior to SP3

XSS – Cross Site Scripting

- Aim: Getting the victim's web browser to execute malicious code
- Many variants. An example:
  - Alice's server hosts an innocent web forum

XSS – An Example