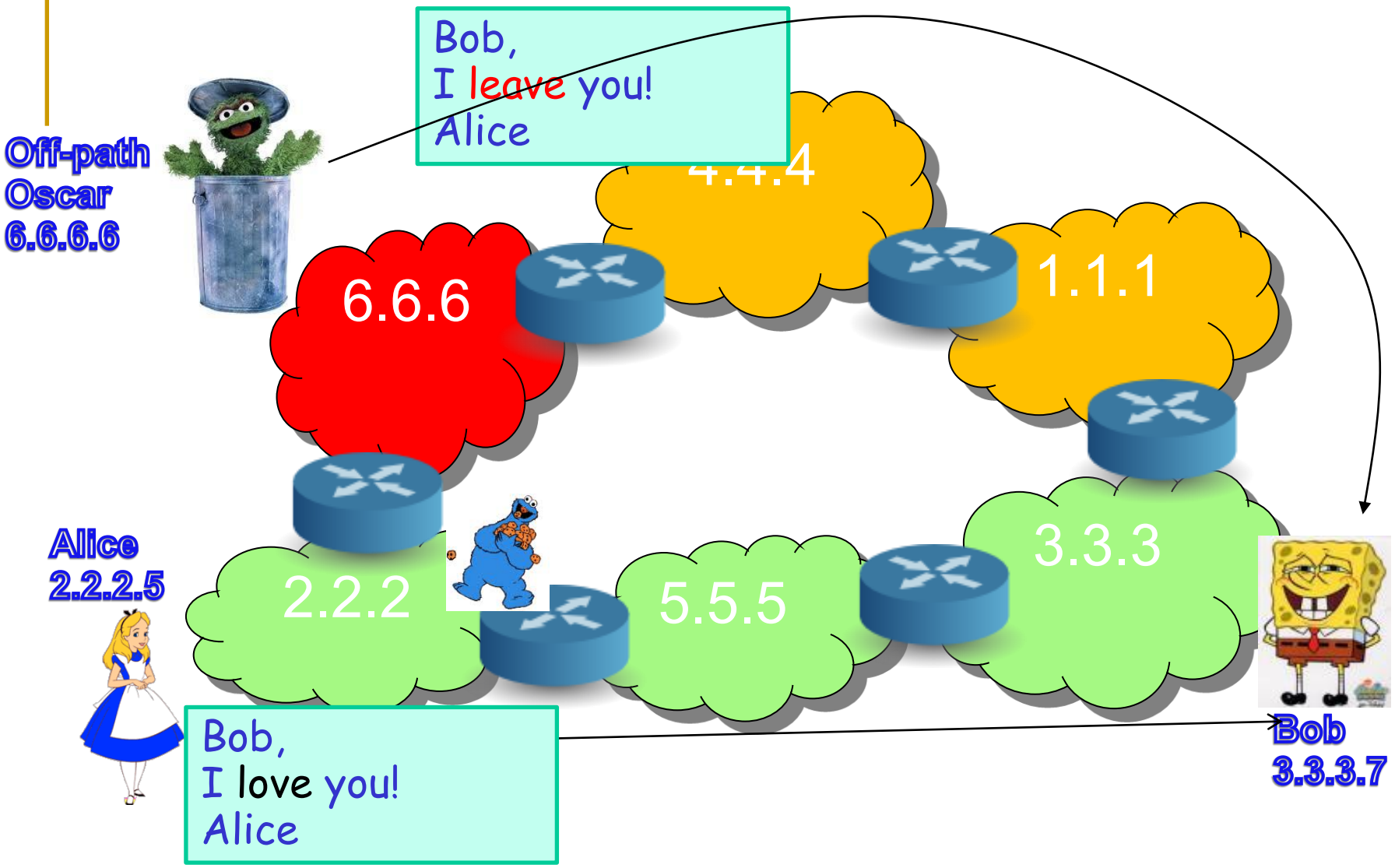
Off-Path Attacking the Web

Yossi Gilad and Amir Herzberg
Computer Science Department, Bar Ilan University

WOOT'12 presentation

Oscar: the Off-Path Attacker



Why Off-Path Attacks?

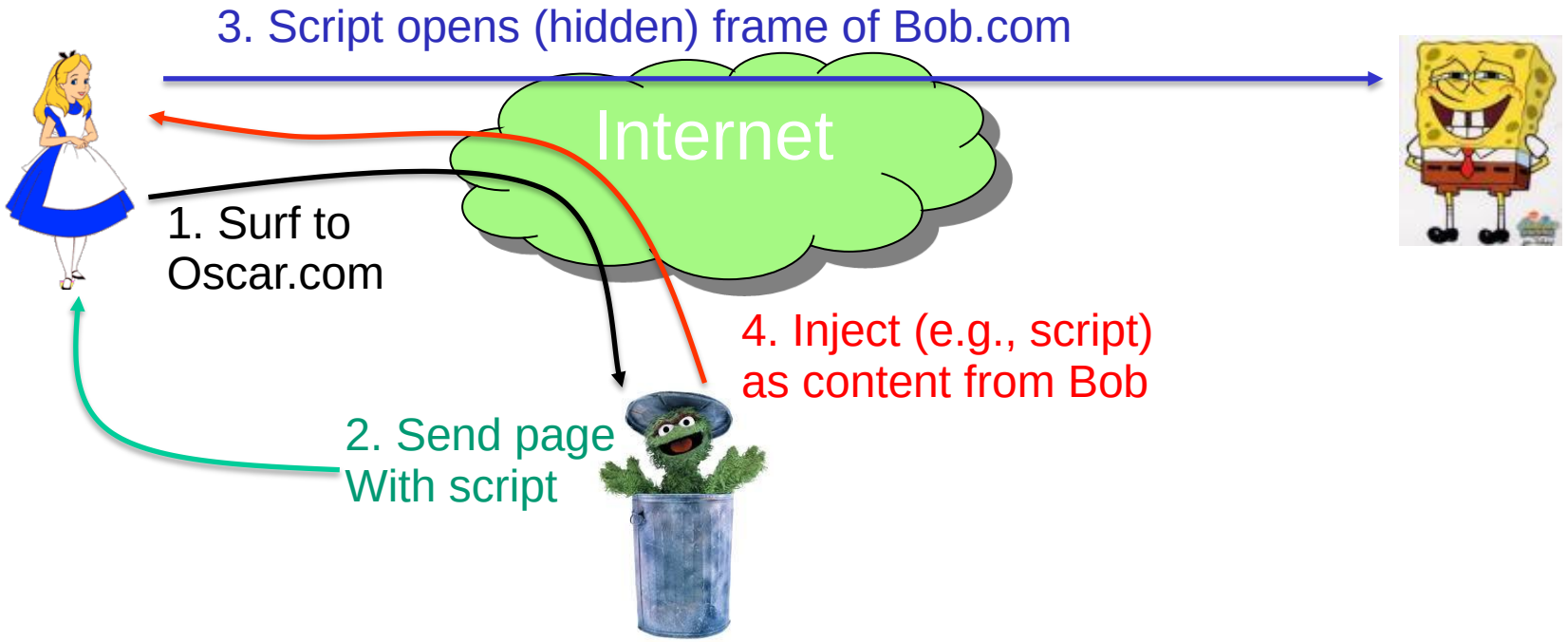
- Why not MitM, Eavesdropping?
  - Harder: physical access or control sw/router
- Can Oscar spoof IP packets?
  - Often not: most ISPs ingress-filter
  - But enough ISPs don't… not so easy to filter
- What of challenge-response like TCP, DNS?
  - Correct use of challenge-response suffices
  - But: Often, challenge-response **used incorrectly**
    - **Since used for other purposes, e.g., SEQ/ACK**
  - This work: **Off-path Web-site Injection**
    - **Allows XSS, phishing and more…**

Related Works

- (Off-path) TCP injections:
  - Predictable ISNs: Morris85, Mitnick95, Zalewski01,05
    - Address-based client authentication vulnerable [Bellovin89]
  - `PoC` for Windows clients: klm07
    - We improve (FW, efficiency), extend to exploit
  - QianMao12, QMXie12: (limited) malware
    - QM12: Also assumes seq#-checking-fw
    - And: only learns server seq# ➔ can't inject to Windows
- Other off-path attacks (not injections)
  - TCP & Tor traffic analysis: GiladH12
  - DNS poisoning: Kaminsky08; H+Shulman12
  - IP packet intercept, modify and kill: GiladH11

Attack Goal and Scenario

1. Alice surfs to Oscar's site
2. Alice's browser runs Oscar's script (puppet)
3. Puppet sends requests to Bob
4. Attacker injects into connection
   - E.g., sends script to Alice, spoofing as Bob

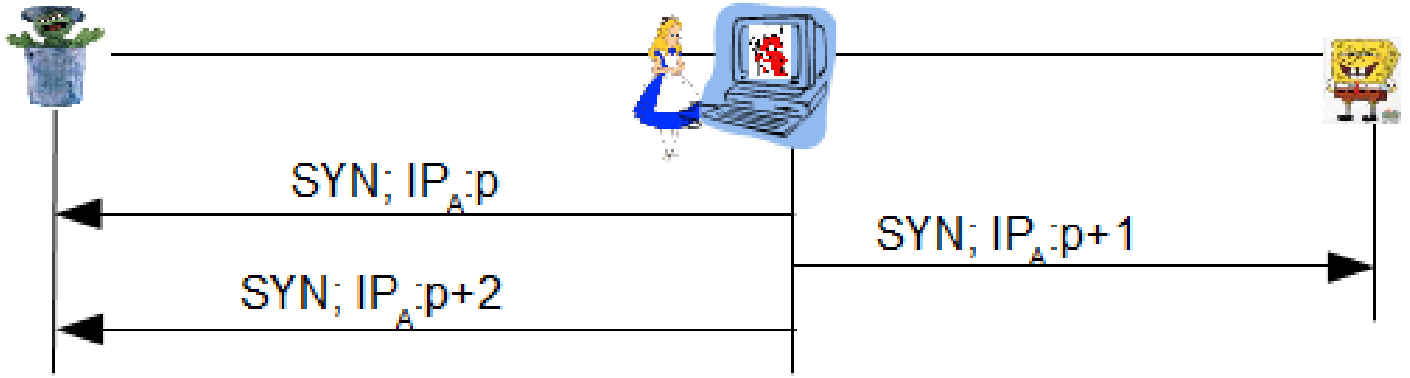


Attack and Talk Overview

- Learn connection identifiers (IPs:ports)
- Learn server's sequence number
- Learn client's sequence number
- Exploit(s):
  - XSS
  - CSRF
  - Phishing
- [Defenses and conclusions]

Learning connection identifiers

- Identifiers: <srcIP:srcPort, dstIP:dstPort>
- Puppet opens connection to Bob (server)
  - ServerIP:port selected by puppet (attacker)
  - Client IP: known from client connection to Oscar
- Client port: sequentially assigned… [Windows,…]



- Not sequential? Test all (cf. [GiladH12])

Finding Server SEQuence Number

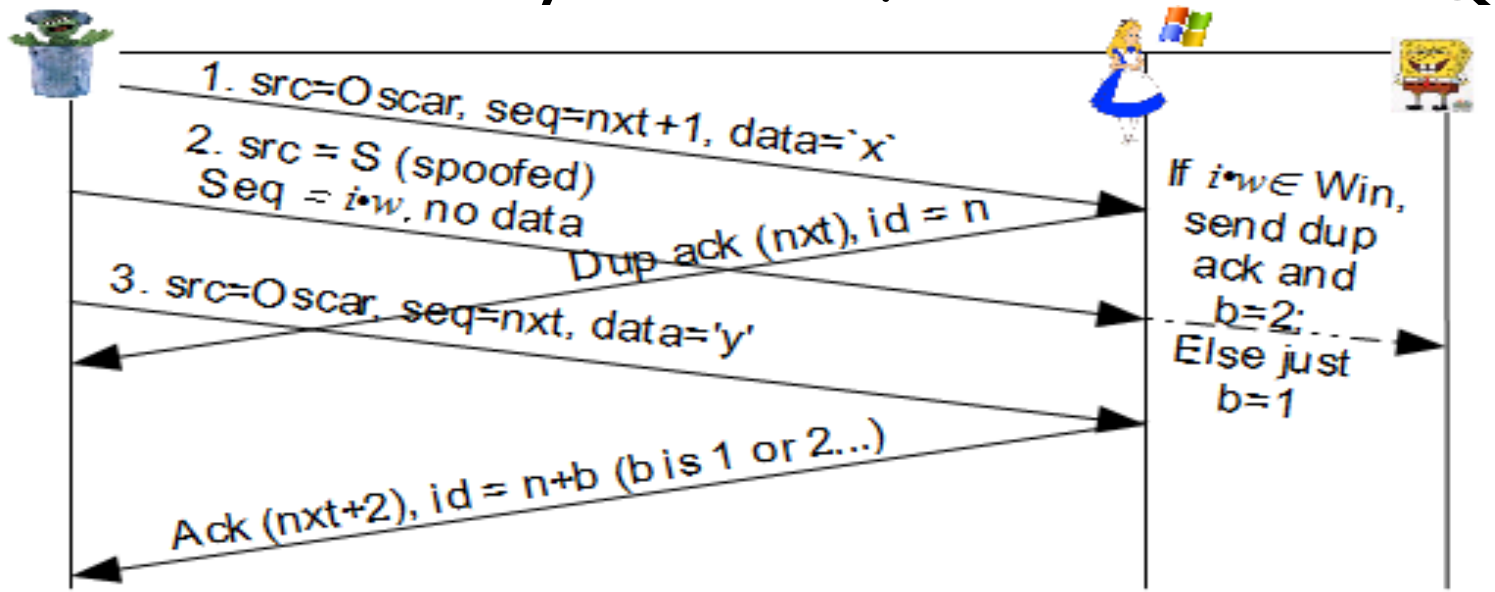
- How? Use TCP responses to *probe* packets
- Empty-ack packets provide useful response:
  - If SEQ out of WIN: send ACK (to re-sync)
  - If SEQ is within WIN: no response (to avoid `storm')
- How to detect if response is sent?
  - Use IP-ID side channel!
  - IP-ID: 16 bit identifier in IP header
    - Used to correctly reconstruct packet from fragments
    - In Windows: globally- incrementing counter
    - One connection (to attacker) leaks info about another!

    - Old trick: NMAP's idle-scan, Bellovin machine-count,…

Finding Server SEQuence Number

1. Puppet opens connection to server
2. Oscar sends query-probe-query:
   1. Query: unordered 1-byte packets → ACK (ipid)
   2. Probe (srcIP:server): empty-Ack with SEQ=$i \cdot w$
      - w is estimate of WIN size

- Found → binary search finds exact SEQ !!

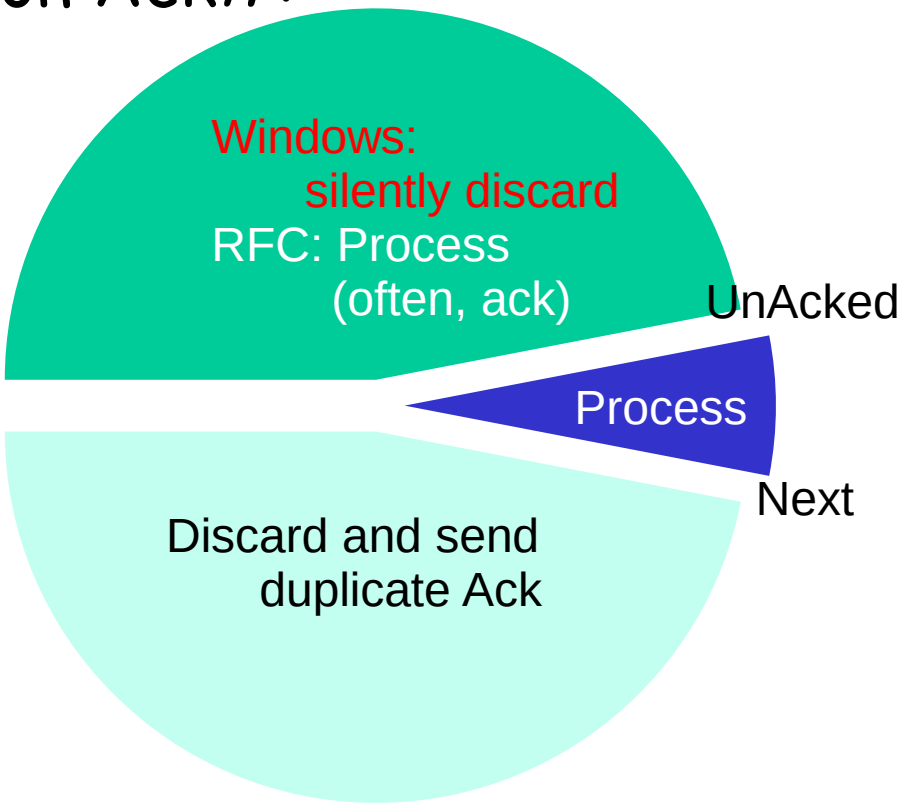


Attack and Talk Overview

- Puppet opens connection to server
  - Known IPs and server port
- Learn connection identifiers (client port)
- Learn server's sequence number
- Learn client's sequence number
- Exploit(s):
  - XSS
  - CSRF
  - Phishing
- [Defenses and conclusions]

Finding **Client** SEQuence Number

- We already know server seq (and IPs, ports)
- To find client seq#: send pkt w/ data
  - With server's IP:port, correct seq#
  - TCP's handling depends on Ack#:
- For Windows clients:
  - As of XP SP2
  - Silently discards pkt with `old` ack number
  - Otherwise: send ACK
- Leaks: Ack#>UNA
- Binary search…

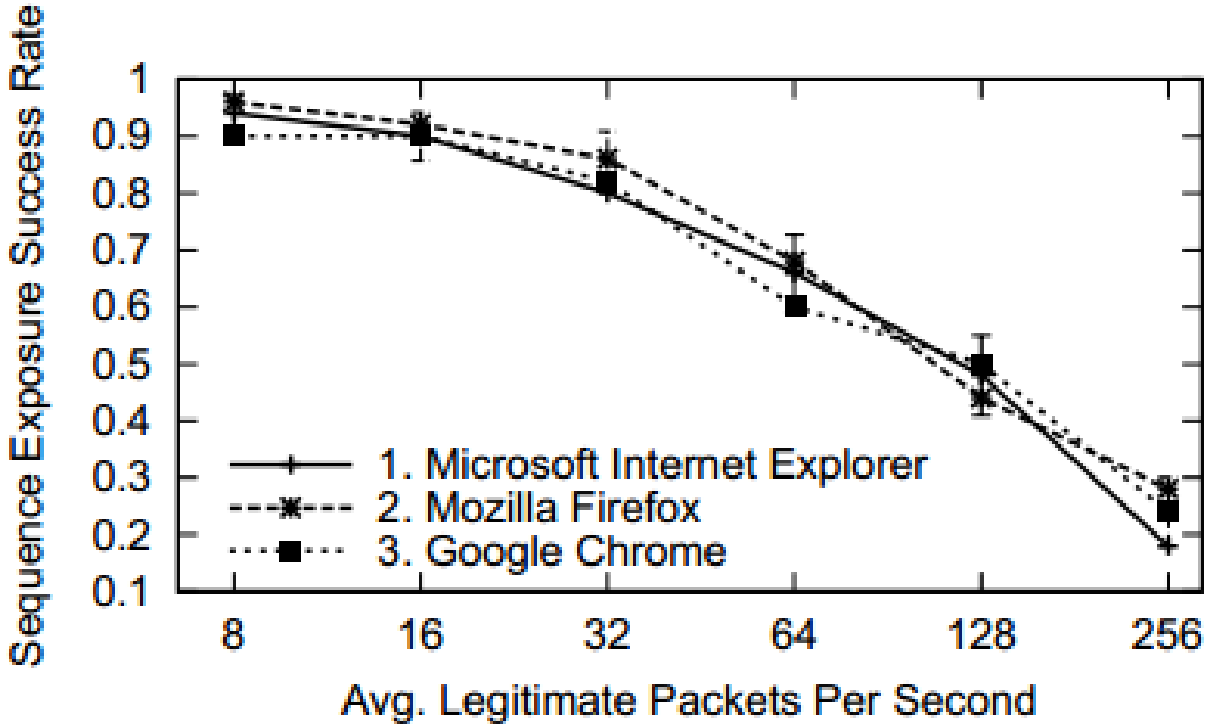


TCP Injection: Challenges

- Firewall passing: Ok
- Lost probes: double-check `no-ack` events
- Lost query/answer: detect via TCP's Acks
- Irrelevant packet sent (IP-ID incremented): repeat `suspect tests'
- Not too many extra checks (or failures)…
  - When in doubt, read the paper!
- Results…

TCP Injection: Success Rates

- Scenario:
  - Apache server, Windows clients, 10Mbps
  - Attacker: 1Mbps; RTT to client: 100msec
  - Avg. time: 102sec [std deviation: 18sec]

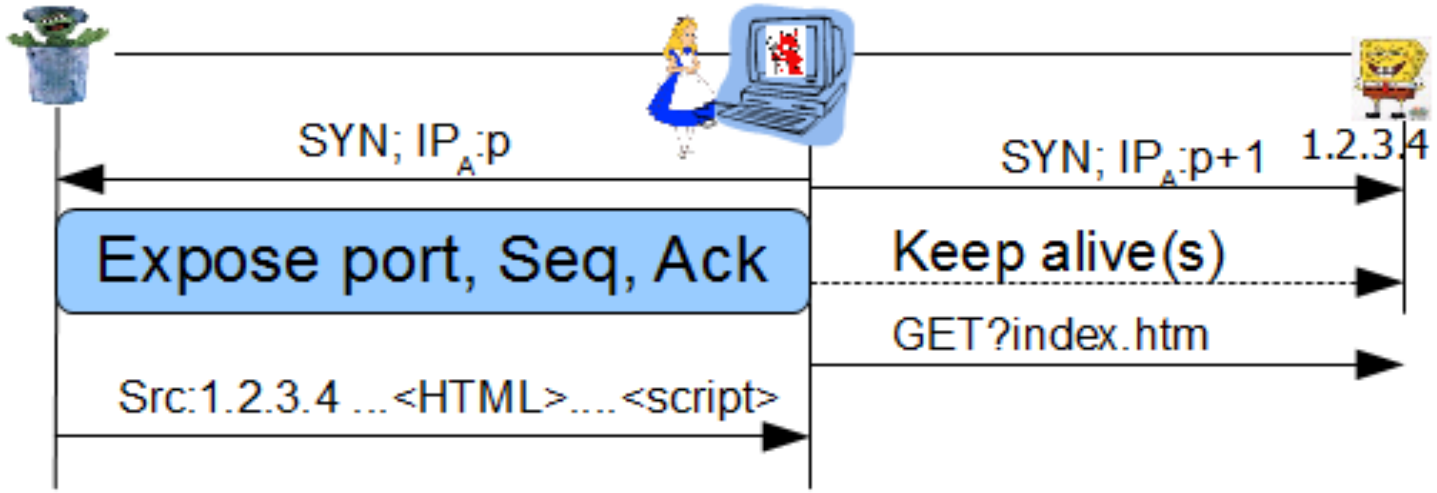


Attack and Talk Overview

- Puppet opens connection to server
  - Known IPs and server port
- Learn connection identifiers (client port)
- Learn server's sequence number
- Learn client's sequence number
- Exploit(s):
  - XSS
  - CSRF
  - Phishing
- [Defenses and conclusions]

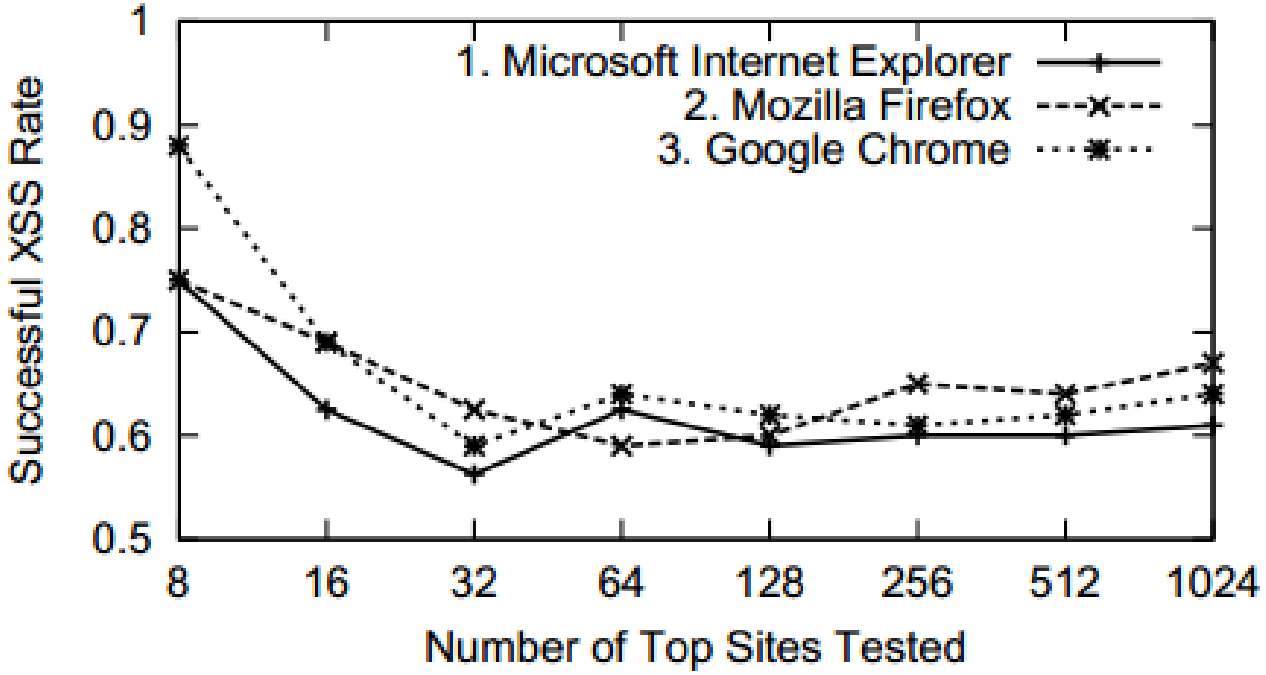
Exploiting Injections: XSS, CSRF

- Cross Site Scripting (XSS): cause browser to run MalScript in context of victim.com
  - Known XSS: exploit bug in site or browser
  - Off-path-injected XSS: no need for vulnerable site/browser!
  - Can post fake requests – like CSRF, but…
  - **Circumvents**: SOP, *origin* header, *CSP, referrer…*



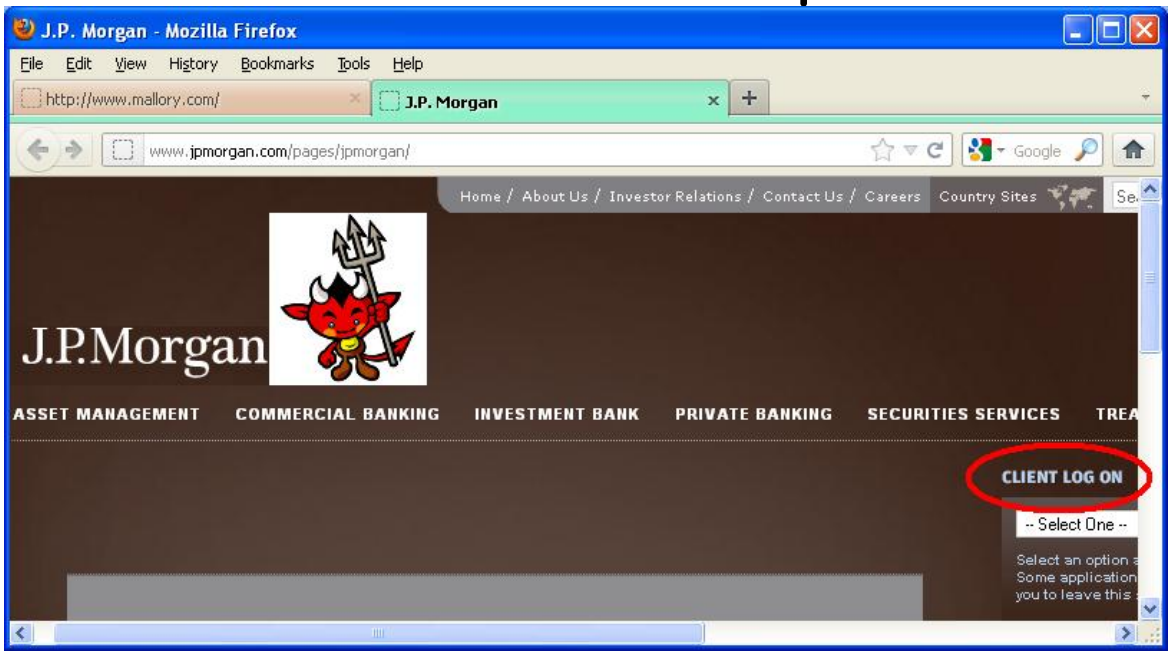
XSS Exploit: Results

- Top 1024 sites, 10Mb win clients, 1Mb Oscar
- Avg 32 pkts/s `noise`
- Immune sites: mostly SSL or non-persistent



Phishing by Injection

- Off-path XSS, CSRF may fail:
  - To collect user-entered data, e.g., passwords
  - Esp. if site uses SSL for passwords
- Alternative: phish / deface !
  - Change contents: steal PWDs, push malware…



Phishing by Injection

- Off-path XSS, CSRF may fail:
  - To collect user-entered data, e.g., passwords
  - Esp. if site uses SSL for passwords
- Alternative: phish / deface !
  - Change contents: steal PWDs, push malware…
- Spoof page **only** when user asks for it
  - Puppet maintains open connection
  - Detect user requesting victim page
    - By detecting increase in client-seq-number
  - `Kill` real response from server
    - Send data with server's SEQ in advance

Defenses and Conclusions

- Defenses
  - Client: Use unpredictable IP-ID, ports
    - Not random… see paper for details
  - Server / FW: drop connections with too many suspect (empty) Acks
- Conclusions
  - TCP may not be secure against off-path !
    - SOP is not much better than client address auth!
    - **Use `real' security: SSL/TLS, IPsec**, etc.
  - Attacks may be improved, abused further…

Thank You!

- Questions?
- Demo??