如何”黑”掉美国国家安全局(NSA)网站
--Host of Troubles攻击

陈建军

导师：段海新教授

Host of Troubles: Multiple Host Ambiguities in HTTP Implementations

Jianjun Chen, Jian Jiang, Haixin Duan,

Nicholas Weaver, Tao Wan, Vern Paxson

一个Host-of-Troubles攻击演示

一个Host-of-Troubles攻击演示

squid-cache.org
Optimising Web Delivery

docs | download | donate | support | about | contact | shop | blog

Introduction

- About Squid
- Why Squid?
- Squid Developers
- How to Donate
- How to Help Out
- Getting Squid
- Squid Source Packages
- Squid Deployment Case-Studies
- Squid Software Foundation

Documentation

Configuration:

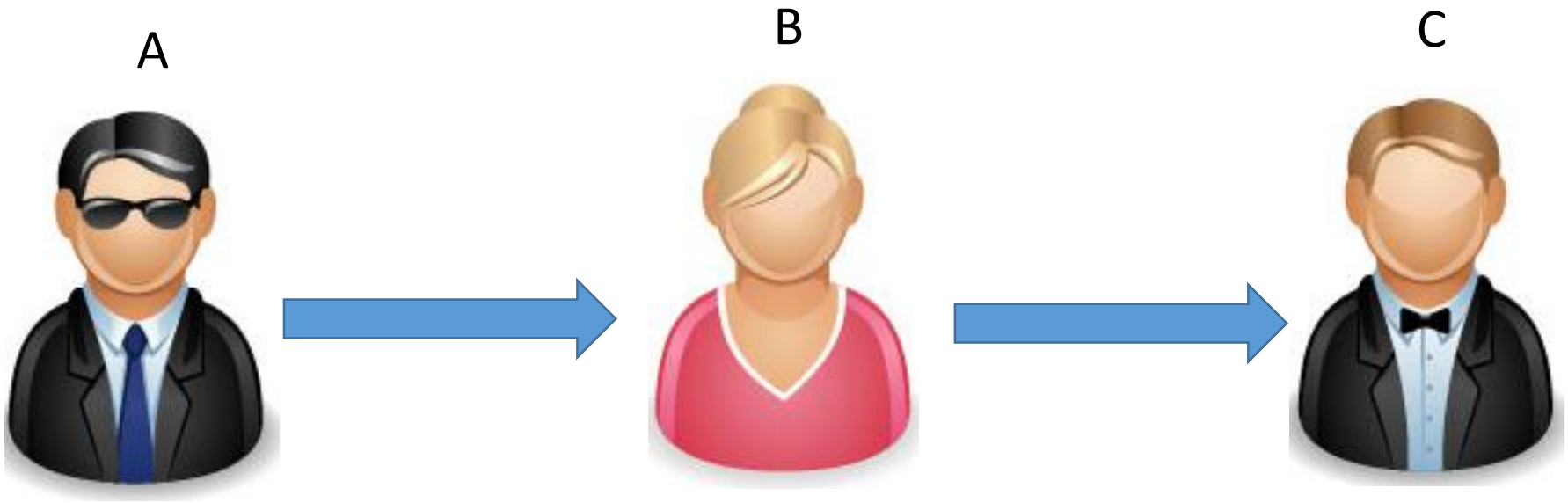
- Reference
- Examples

FAQ and Wiki

Squid Advisories

SQUID-2016:9 (CVE-2016-4555, CVE-2016-4556), May 06, 2016
Fixed from 4.0.10, 3.5.18
Multiple Denial of Service issues in ESI Response processing.
SQUID-2016:8 (CVE-2016-4554), May 06, 2016
Fixed from 3.5.18
Header smuggling issue in HTTP Request processing.
SQUID-2016:7 (CVE-2016-4553), May 06, 2016
Fixed from 4.0.10, 3.5.18
Cache poisoning issue in HTTP Request handling.
SQUID-2016:6 (CVE-2016-4052, CVE-2016-4053, CVE-2016-4054), Apr 20, 2016
Fixed from 4.0.9, 3.5.17
Multiple issues in ESI processing.
SQUID-2016:5 (CVE-2016-4051), Apr 20, 2016
Fixed from 4.0.9, 3.5.17
Buffer overflow in cachemgr.cgi.
SQUID-2016:4 (CVE-2016-3948), Apr 02, 2016
Fixed from 4.0.8, 3.5.16
Denial of Service issue in HTTP Response processing.
SQUID-2016:3 (CVE-2016-3947), Apr 02, 2016
Fixed from 4.0.8, 3.5.16
Buffer overrun issue in pinger ICMPv6 processing.
SQUID-2016:2 (CVE-2016-2569, CVE-2016-2570, CVE-2016-2571, CVE-2016-2572), Feb 23, 2016
Fixed from 4.0.7, 3.5.15
Multiple Denial of Service issues in HTTP Response processing.
SQUID-2016:1 (CVE-2016-2390), Feb 16, 2016
Fixed from 4.0.6, 3.5.14
Remote Denial of service issue in SSL/TLS processing
SQUID-2015:3, Sep 17, 2015
Fixed from 3.5.9

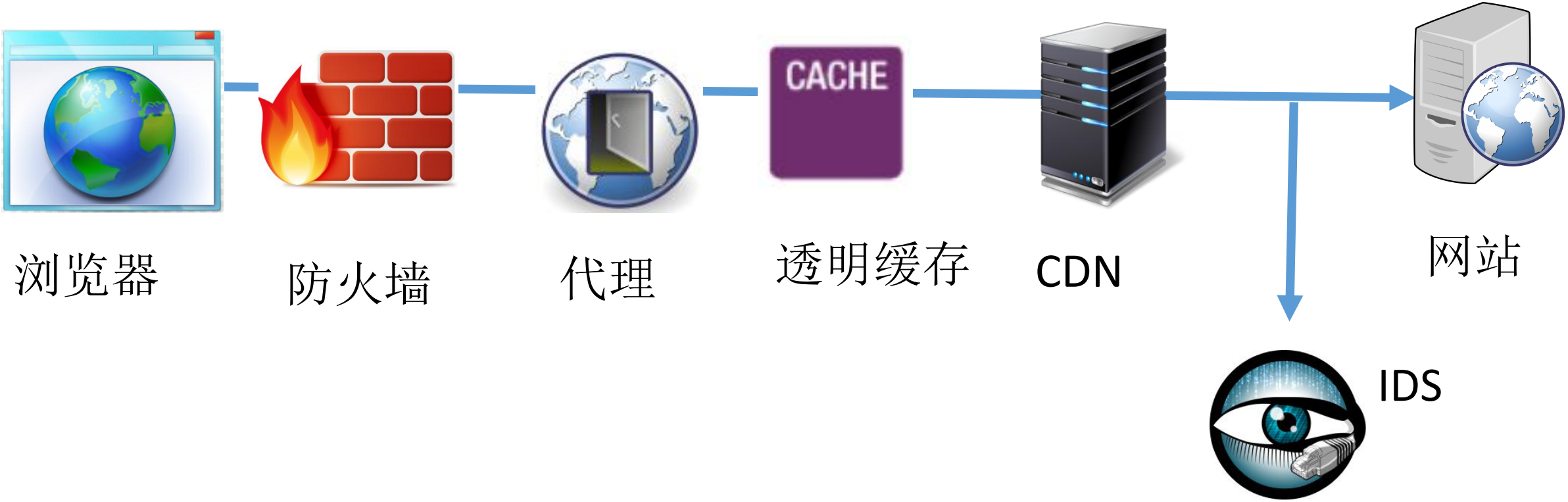
多方理解歧义的例子



我已出发三天即到　　*我已出发，三天即到*　　*我已经出发三天，即到*

生活中，多方对歧义语句理解不一致，可能引起误会

当前Web环境下复杂的多方交互



不同两方之间对消息解析理解不一致，就有可能造成安全问题

预备知识

HTTP请求解析流程

文本消息

GET / HTTP/1.1\r\nHost: a.com\r\nUser-Agent:Mozilla…

Parse

协议字段

GET / HTTP/1.1
host a.com
user-agent Mozilla
…

Interpret

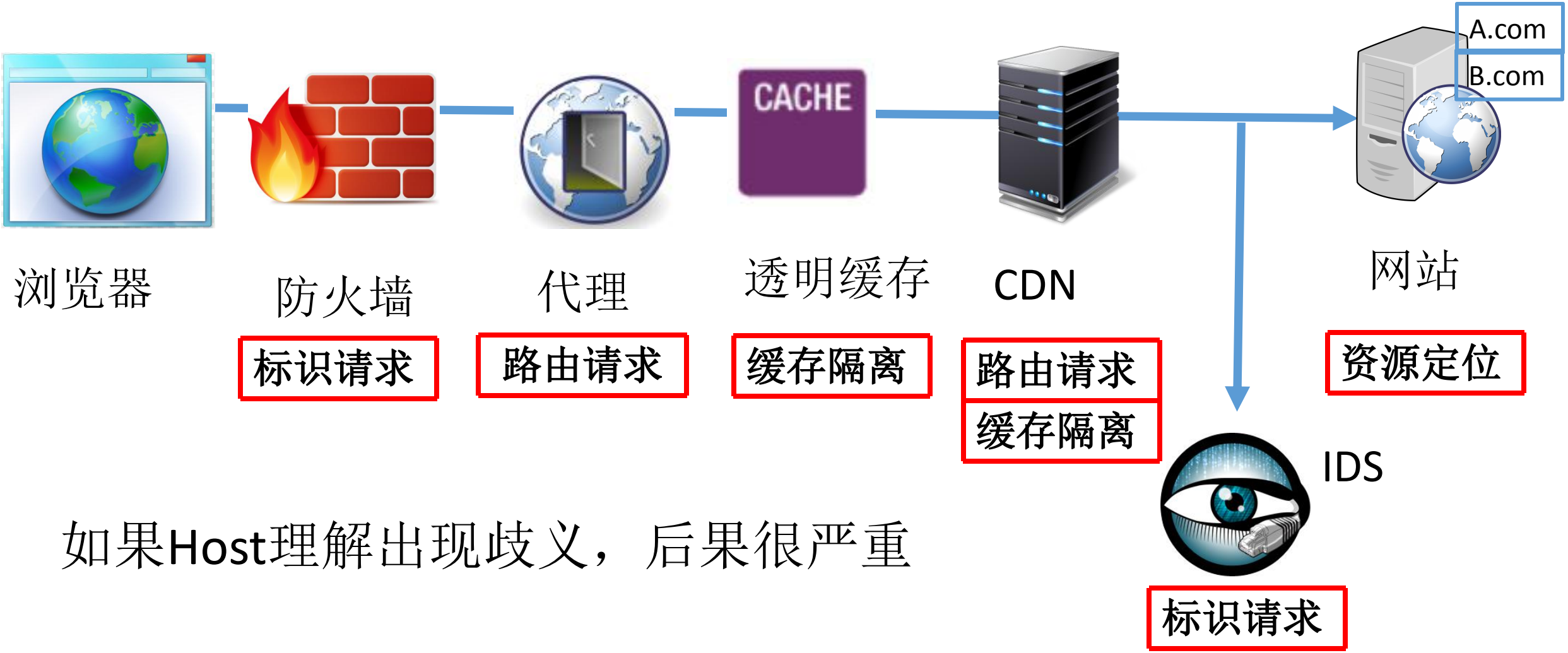
语义结构

```
typedef struct {
    ngx_list_t        headers;

    ngx_table_elt_t   *host;
    ngx_table_elt_t   *user_agent;
    ngx_table_elt_t   *referer;
    ngx_table_elt_t   *content_length;
    ngx_table_elt_t   *content_type;
```

后续操作

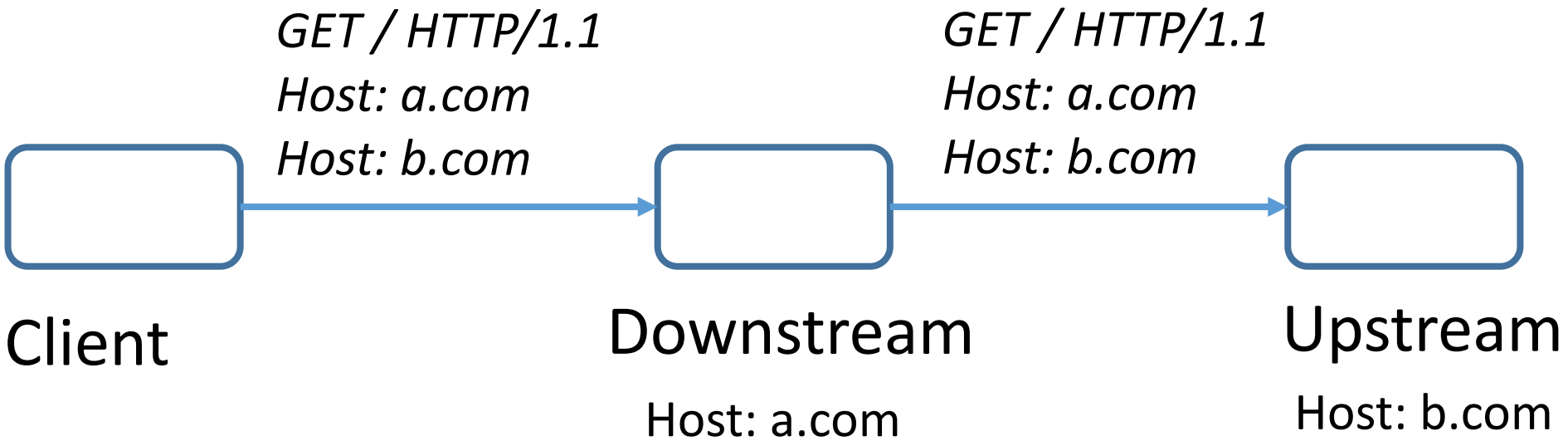
HTTP协议中最关键的字段--Host



如果Host理解出现歧义，后果很严重

如何让不同系统对Host理解产生歧义？

技巧 1: 多个不同Host 头



HTTP标准（HTTP/1.1）

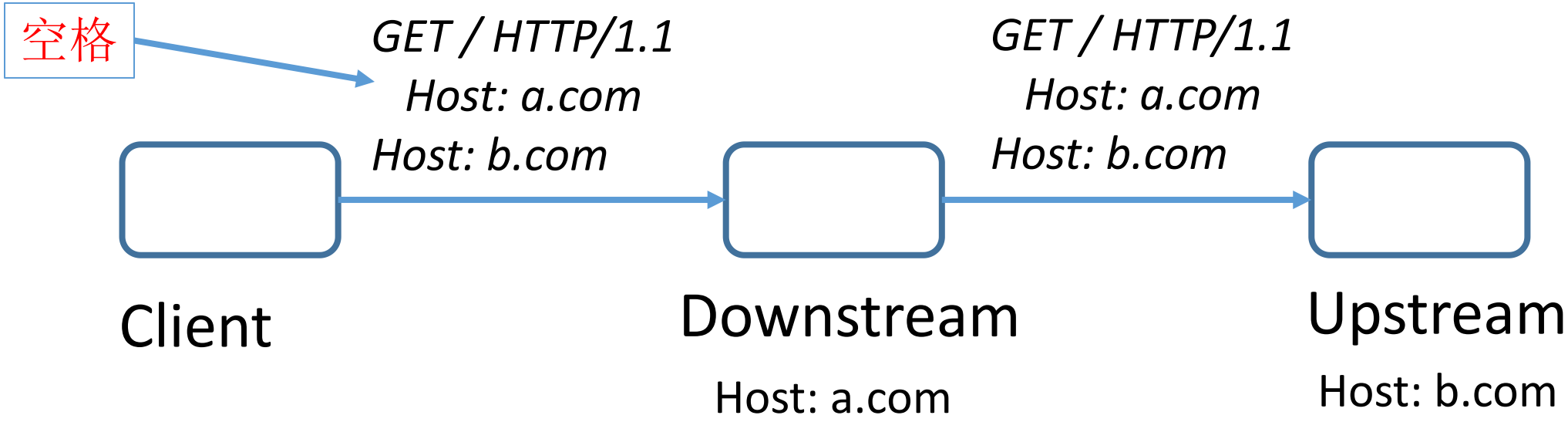
- RFC 2616 隐式要求拒绝多Host头
- RFC 7230 明确要求拒绝多Host头

不同系统对多Host头请求的处理

Implementation		Preference	Implementation		Preference	Implementation		Preference
Server	Apache	Concatenate	CDN	Akamai	First	Firewall	Bitdefender	First
	IIS	Reject		Alibaba	First		ESET	Last
	Nginx	First		Azure	Reject		Huawei	First
	Tomcat	First		CloudFlare	First		Kaspersky	First
Transparent cache	ATS	First		CloudFront	First		OS X	Concatenate
	Squid	First		Fastly	Reject		PAN	First
Reverse Proxy	Nginx	First		Level 3	First		Windows	First
	Varnish	Reject		Tencent	Last			

绝大部分系统不遵循RFC7230， 不同系统之间出现歧义

技巧 2： Host头增加前后空格



HTTP标准

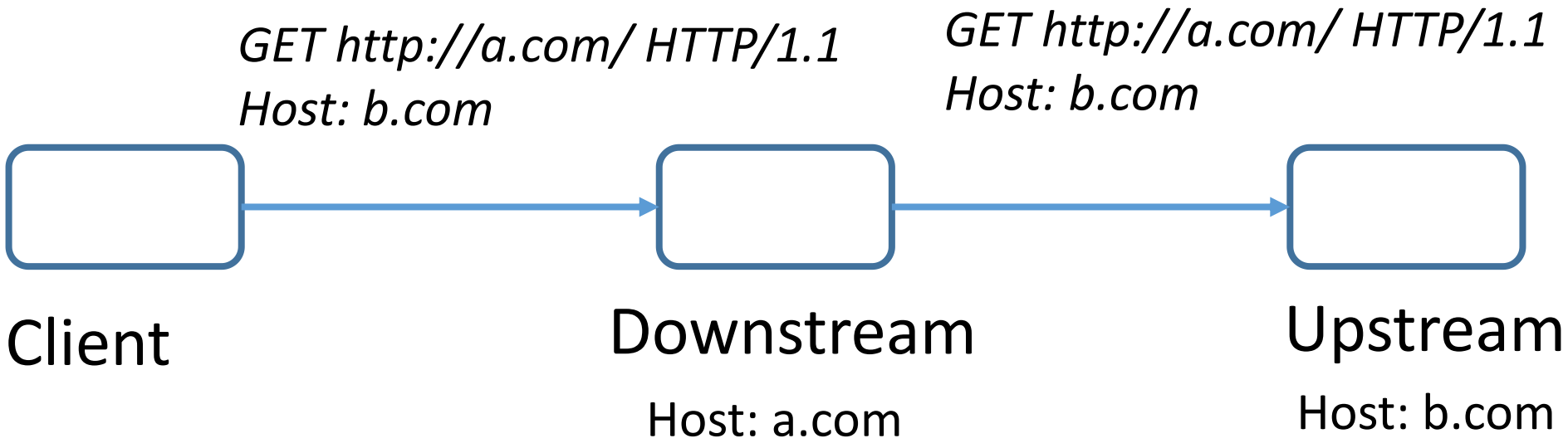
	首个头为空格前置Host	其他空格前置Host头	空格后置Host头
RFC 2616	Reject (implicit)	Line folding	Recognize
RFC 7230	Reject	Reject	Reject

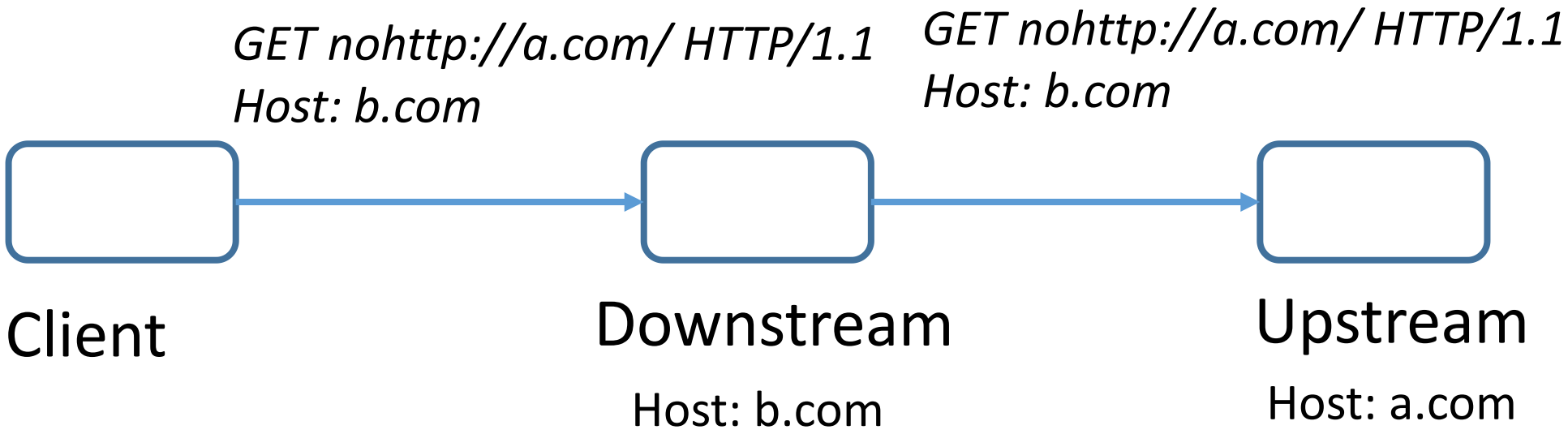
不同系统对空格Host的处理

		首个前置空格	其他前置空格	后置空格
Server	Apache	Not recognize	Line folding	Recognize
	IIS	Recongize	Line folding	Recognize
	Nginx	Not recognize	Not recognize	Not recognize
Transparent Cache	ATS	Not recognize	Not recognize	Not recognize
	Squid	Recongize	Recongize	Recongize
CDN	Akamai	Recongize	Recongize	Recongize
	Alibaba	Not recognize	Not recognize	Not recognize
	CloudFlare	Not recognize	Not recognize	Not recognize
	Tencent	Recongize	Recongize	Recongize
Firewall	Huawei	Not recognize	Not recognize	Not recognize
	PAN	Not recognize	Not recognize	Not recognize

绝大部分系统不遵循RFC, 各个系统理解歧义差别很大！

技巧 3： Request-URI是绝对路径





技巧 3： Request-URI是绝对路径

HTTP标准

	Preference	Schema
RFC 2616	Absolute-URI	Not specified
RFC 7230	Absolute-URI	Not specified

HTTP实现

- 当Absolute-URI 与Host同时出现时：
  - 除了Akamai，绝大多数系统遵循RFC

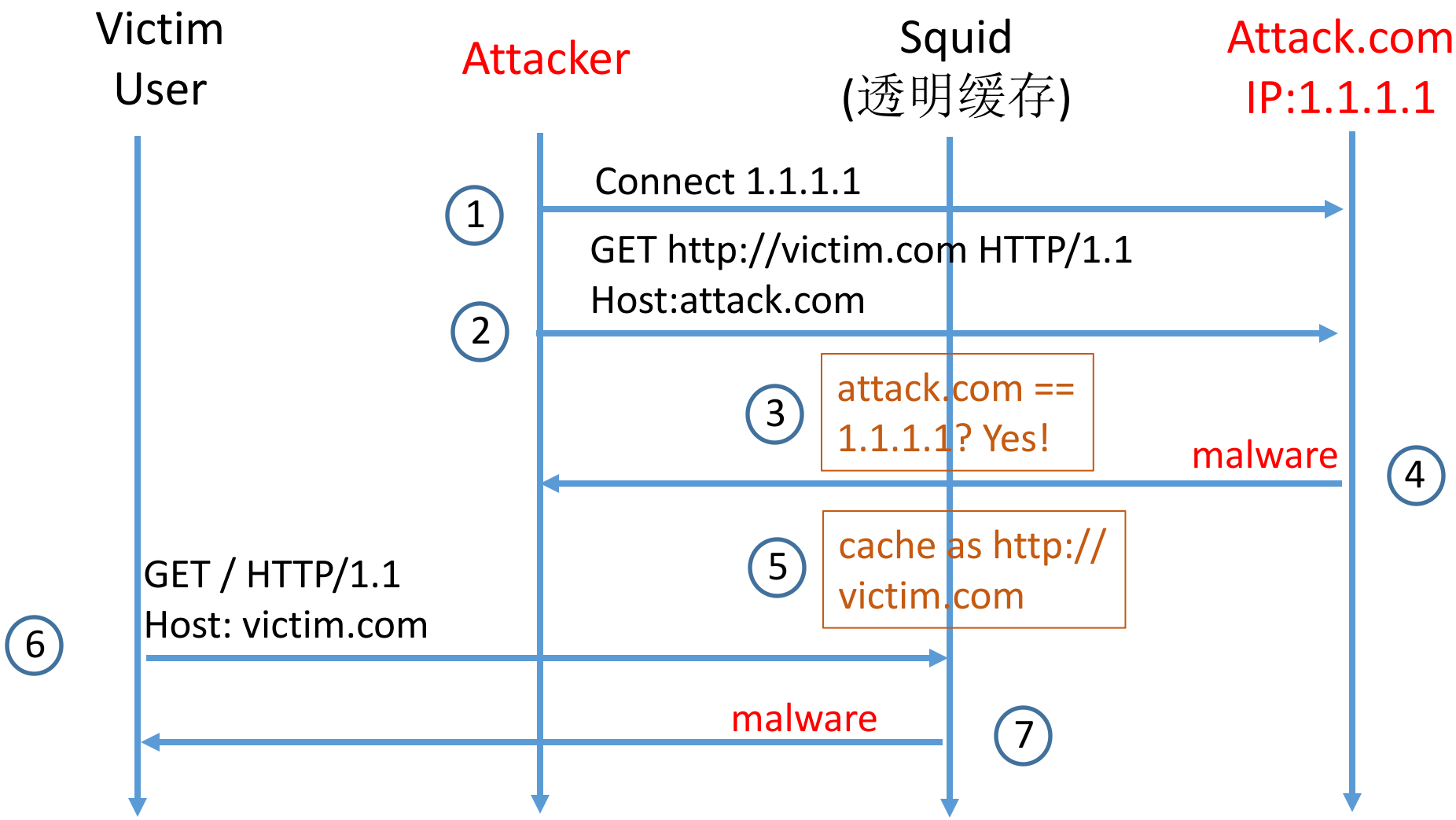
不同系统对绝对URI歧义的处理

Implementation		Schema	Implementation		Scheme	Implementation		Scheme
Server	Apache	HTTP only	CDN	Akamai	HTTP/S	Firewall	Bitdefender	Fail-open
	IIS	HTTP/S		Alibaba	any		ESET	any
	Nginx	any		Azure	HTTP/S		Huawei	any
	Tomcat	HTTP/S		CloudFlare	any		Kaspersky	any
Transparent cache	ATS	any		CloudFront	any		OS X	HTTP only
	Squid	HTTP only		Fastly	HTTP only		PAN	HTTP/S
Reverse Proxy	Nginx	any		Level 3	HTTP/S		Windows	any
	Varnish	HTTP only		Tencent	HTTP only			

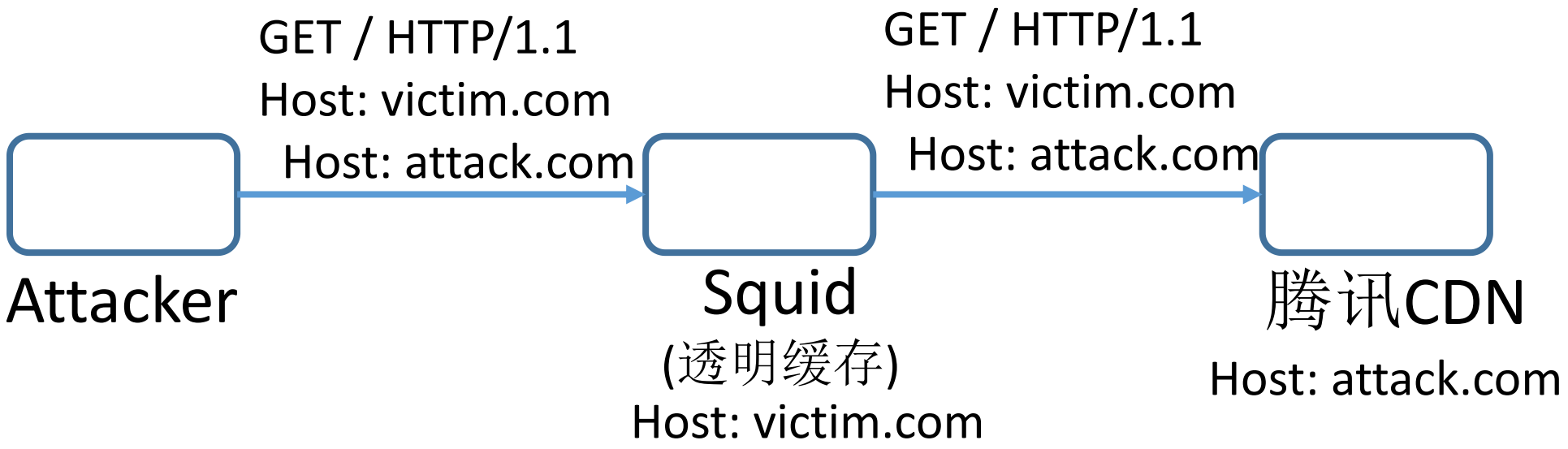
各个系统之间理解歧义再次增大！

如何利用Host歧义造成攻击？

例 1: 任意网站Squid透明缓存污染(毒鱿鱼攻击)

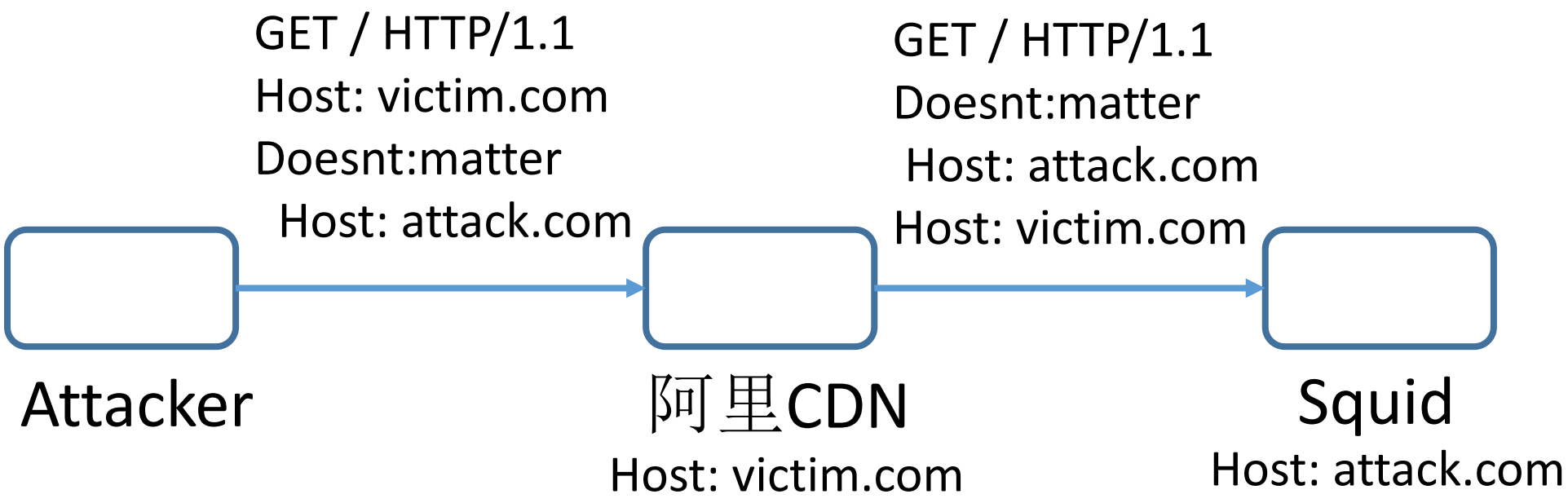


例 2: Co-hosting网站透明缓存污染

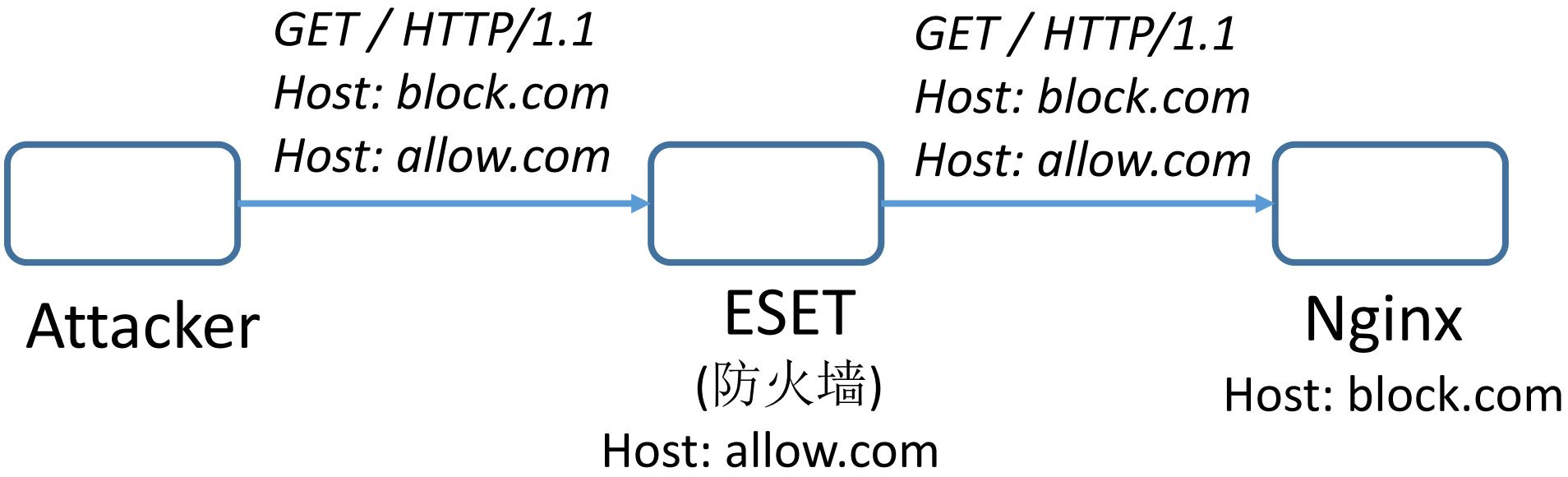


Downstream 还可以是Apache Traffic Server
Upstream 还可以是Akamai 等

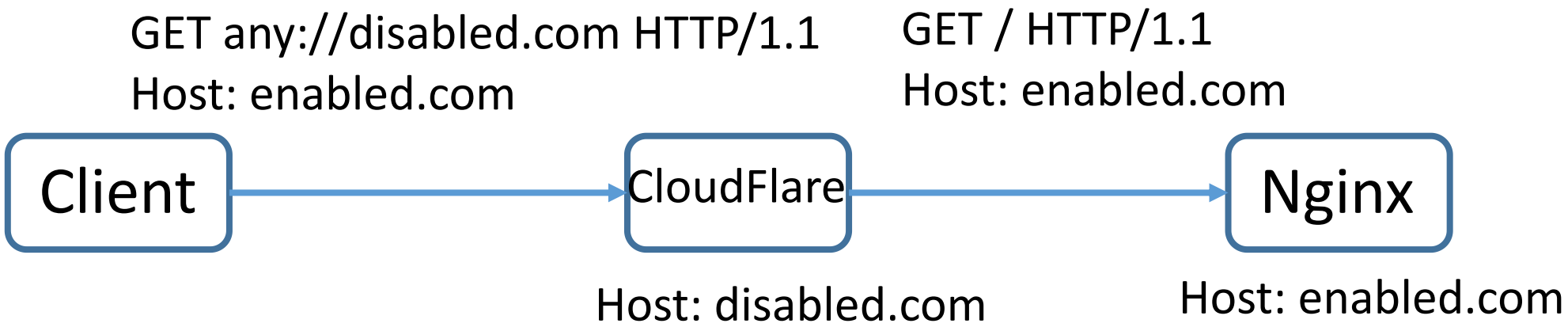
例 3: CDN缓存污染

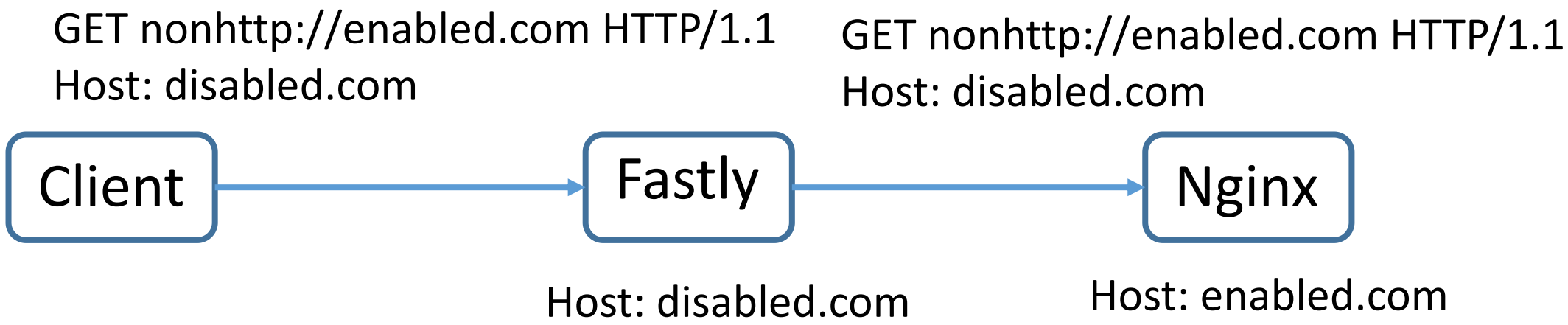

GET / HTTP/1.1
Host: victim.com
Doesnt:matter
Host: attack.com
GET / HTTP/1.1
Doesnt:matter
Host: attack.com
Host: victim.com
Attacker
阿里CDN
Host: victim.com
Squid
Host: attack.com

例 4: 防火墙绕过


GET / HTTP/1.1
Host: block.com
Host: allow.com
GET / HTTP/1.1
Host: block.com
Host: allow.com
Attacker
ESET
(防火墙)
Host: allow.com
Nginx
Host: block.com

例 5:WAF绕过


GET any://disabled.com HTTP/1.1
Host: enabled.com
GET / HTTP/1.1
Host: enabled.com
Client
CloudFlare
Nginx
Host: disabled.com
Host: enabled.com


GET nonhttp://enabled.com HTTP/1.1
Host: disabled.com
GET nonhttp://enabled.com HTTP/1.1
Host: disabled.com
Client
Fastly
Nginx
Host: disabled.com
Host: enabled.com

Downstream-Upstream combinations

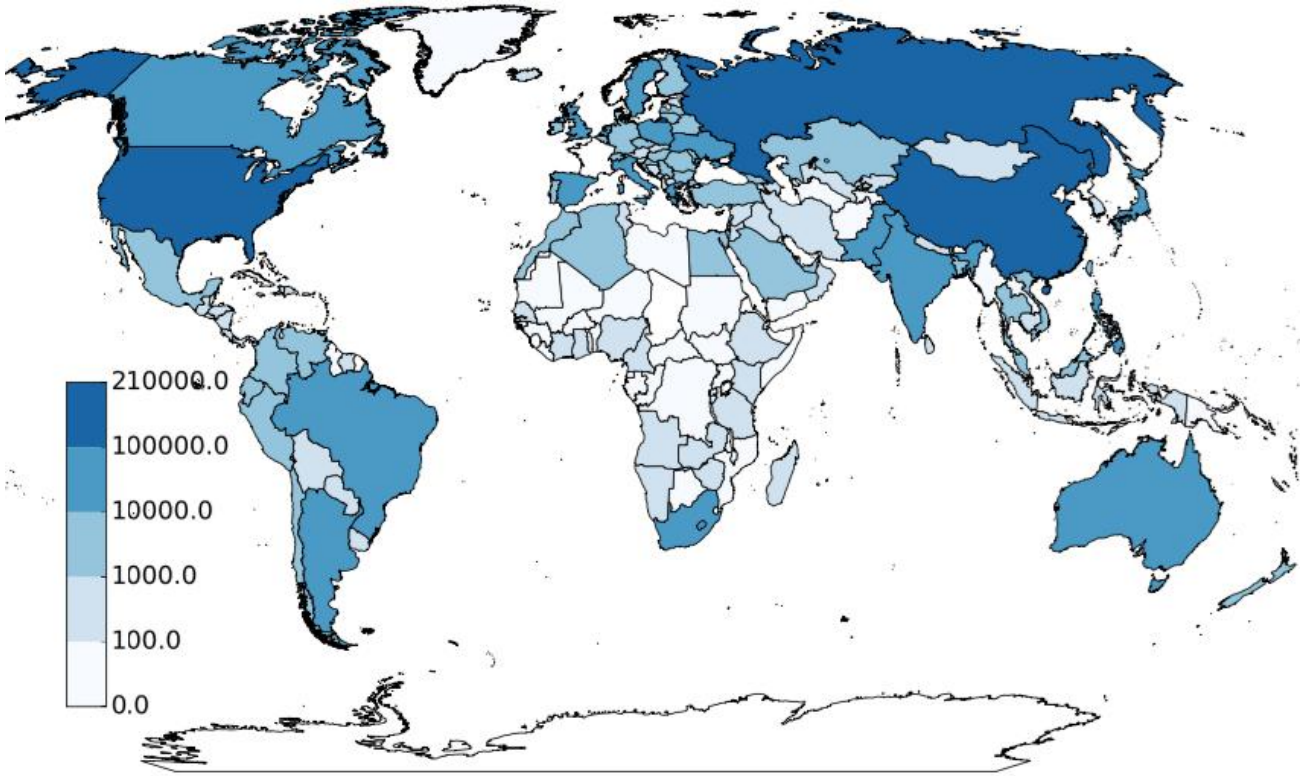
Downstream \ Upstream		Reverse Proxy							CDN								Server					
		Apache	IIS	Lighttpd	LiteSpeed	Nginx	Squid	Varnish	Akamai	Alibaba	Azure	CloudFlare	CloudFront	Fastly	Level3	Tencent	Apache	IIS	Lighttpd	LiteSpeed	Nginx	Tomcat
Transparent Cache	ATS				✓		✓		✓							✓				✓		
	Squid						✓		✓							✓						
Forward Proxy	Apache															✓						
	Squid						✓		✓							✓						
Reverse Proxy	Apache								—	—	—	—	—	—	—	—						
	Lighttpd				✓	✓			—	—	—	—	—	—	—	—				✓	✓	
	LiteSpeed	✓		✓		✓		✓	—	—	—	—	—	—	—	—	✓		✓		✓	✓
	Squid						✓		—	—	—	—	—	—	—	—						
	Varnish		✓	✓	✓	✓			—	—	—	—	—	—	—	—		✓	✓	✓	✓	✓
CDN	Akamai						✓		—													
	Alibaba				✓		✓		✓	—										✓		
	CloudFlare	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	—	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	CloudFront												—			✓						
	Fastly		✓	✓	✓	✓			✓	✓		✓	✓	—	✓			✓	✓	✓	✓	✓
Firewall	Bitdefender	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	ESET	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Huawei	✓	✓	✓	✓		✓	✓	✓		✓			✓		✓	✓	✓	✓	✓		
	Kaspersky	✓	✓	✓	✓		✓	✓	✓		✓			✓		✓	✓	✓	✓	✓		
	OS X	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓
	PAN	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓			✓	✓	✓	✓	✓	✓	✓
	Windows	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

202个可以被利用的歧义组合！

现实世界中ISP缓存有多少被攻击？

测量

- 投放广告
  - Utorrent客户端， 140万展示次数
  - 网站广告，投放20天



测量结果

- Utorrent客户端广告
  - 16,168 个IP 地址发现ISP缓存
  - 其中15,677 (96.9%)个IP地址可以被攻击
- 网站广告
  - 1,376个IP地址发现ISP缓存
  - 1,331 (96.7%) 个IP地址可以被攻击

约97%的ISP缓存用户可以被污染！

Country	ASN	Organization	#
PH	9299	Philippine Long Distance Telephone	2396
IN	23860	Alliance Broadband Service	1234
IN	24309	Atria Convergence Technologies	1013
CN	56046	China Mobile	692
CN	9808	China Mobile	476
PH	132199	Globe Telecom	429
NZ	9790	CallPlus Services Limited	410
NZ	7657	Vodafone NZ Ltd.	377
US	3651	Sprint	317
SA	35819	Etihad Etisalat Company (Mobily)	302

厂商反馈

- 缓存污染
  - Squid： CVE-2016-4553, CVE-2016-4554
  - 腾讯：已经修复
  - 阿里：已经修复
  - Akamai：已经修复
  - Apache Traffic Server：已确认
- 防火墙绕过
  - Palo Alto Networks：增加新选项，已修复
  - 华为：增加新选项，已修复
  - ESET：已修复
  - CloudFlare：已修复
  - Fastly：已修复

如何防御

- 对于厂商，不同的系统应该完全遵循RFC7230，消除与其他系统之间的歧义
  - 拒绝多Host头和包含前后空格Host头的请求
- ISP应该及时更新透明缓存软件
- 对于网站管理员，可以部署https，并启用pre-loaded HSTS来减轻攻击的危害
- 对于普通浏览器用户，可以使用我们的在线工具检测是否受到透明缓存污染攻击。
  - https://hostoftroubles.com/online-checker.html
- 对于其他系统的开发者，应该从这个视角来重新审计他们的实现

讨论

- Jon Postel法则的局限性
  - Be conservative in what you do, be liberal in what you accept from others
- 制定协议标准时，是否可以给出参考代码？
  - 自然语言描述容易出现模糊不清或歧义
- 协议设计时，应该避免出现重复或冲突的字段
  - 而不是在标准规定中澄清
- 如何让不同实现遵循标准？

致谢

paloalto NETWORKS

HUAWEI

Akamai

Tencent

Alibaba.com

eset

CLOUDFLARE

fastly

KASPERSKY lab

traffic server™

谢谢！