“Nuclear” exploit kit service cashes in on demand from cryptoransomware rings

Exploit kit's inner workings exposed as researchers help shut down its servers.

by Sean Gallagher - Apr 22, 2016 6:30am CDT

Share Tweet Email 10

Security researchers at Cisco Talos and Check Point have published reports detailing the inner workings of Nuclear, an "exploit kit" Web service that deployed malware onto victims' computers through malicious websites. While a significant percentage of Nuclear's infrastructure has been recently disrupted, the exploit kit is still operating—and looks to be a major contributor to the current crypto-ransomware epidemic.

Introduced in 2010, Nuclear has been used to target millions of victims worldwide, giving attackers the ability to tailor their attacks to specific locations and computer configurations. Though not as widely used as the well-known Angler exploit kit, it has been responsible for dropping Locky and other crypto-ransomware onto more than 140,000 computers in more than 200 countries, according to statistics collected by Check Point (PDF). The Locky campaign appeared to be placing the greatest demand on the Nuclear pay-to-exploit service.

Much of Talos' data on Nuclear comes from tracking down the source of its traffic—a cluster of "10 to 15" IP addresses that were responsible for "practically all" of the exploit infrastructure. Those addresses were being hosted by a single cloud hosting provider—DigitalOcean. The hosting company's security team confirmed the findings to Talos and took down the servers—sharing what was on them with security researchers.

ars technica

cloud computing & e-crime

cs642

computer security

adam everspaugh
ace@cs.wisc.edu

announcements

* HW4 due in one week
* **This week:** cloud computing and malware & ecrime
* **Next week:** Bitcoin and Android security
* **Friday, May 6:** Exam review session
* **Sunday, May 8:** Final exam

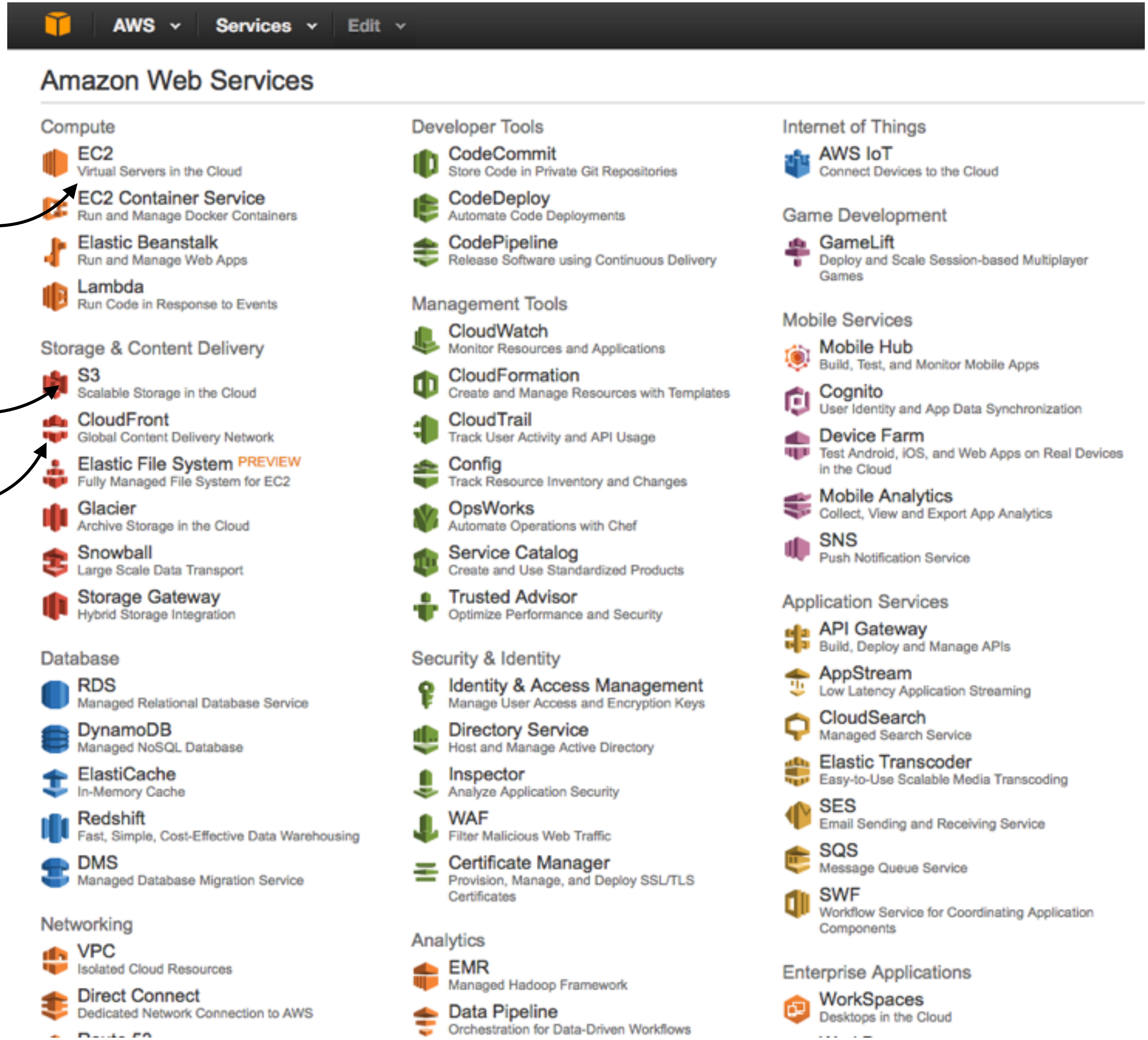
today

* Cloud computing and placement vulnerabilities
* Malware, botnets, and crime

VMs
Infrastructure-as-a-service

Storage

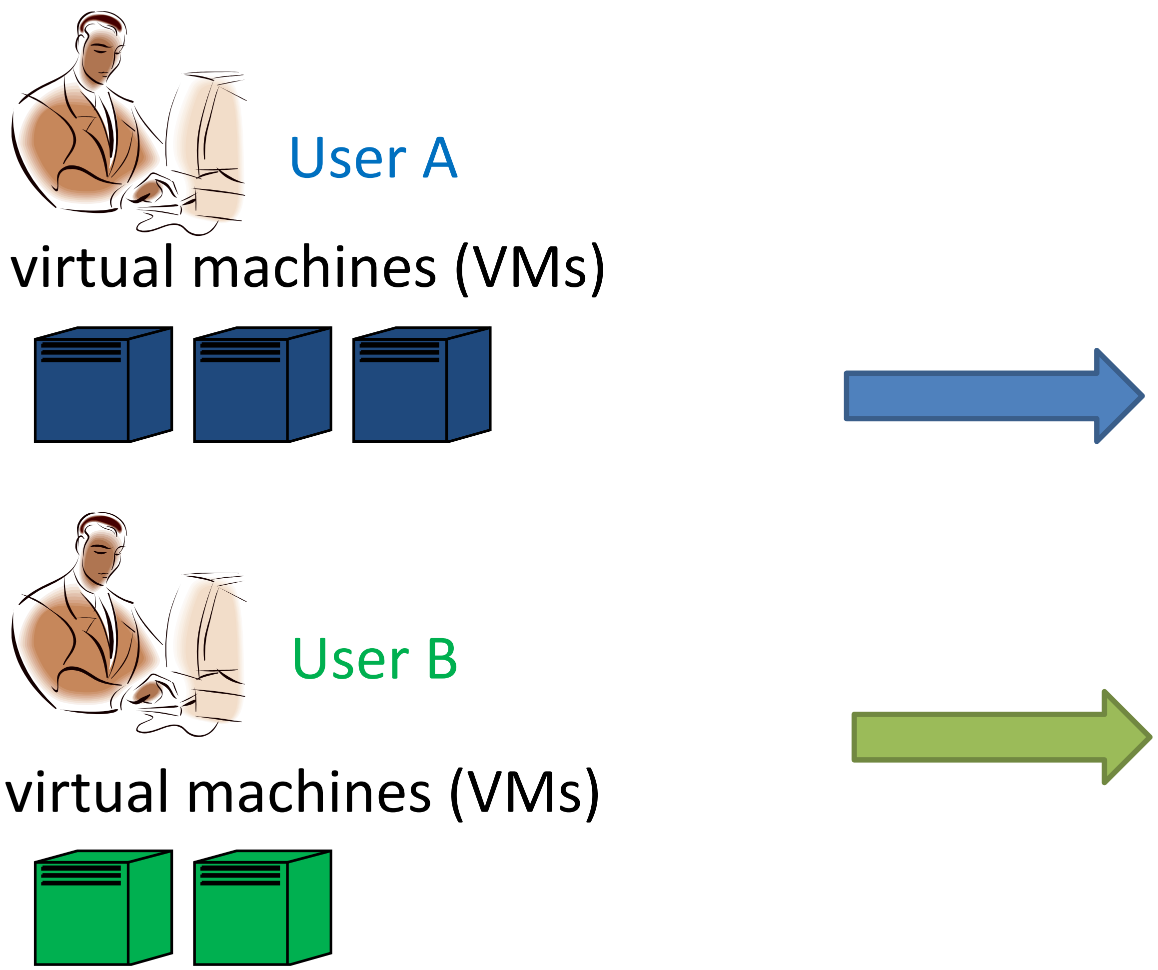
Web Cache/TLS Termination

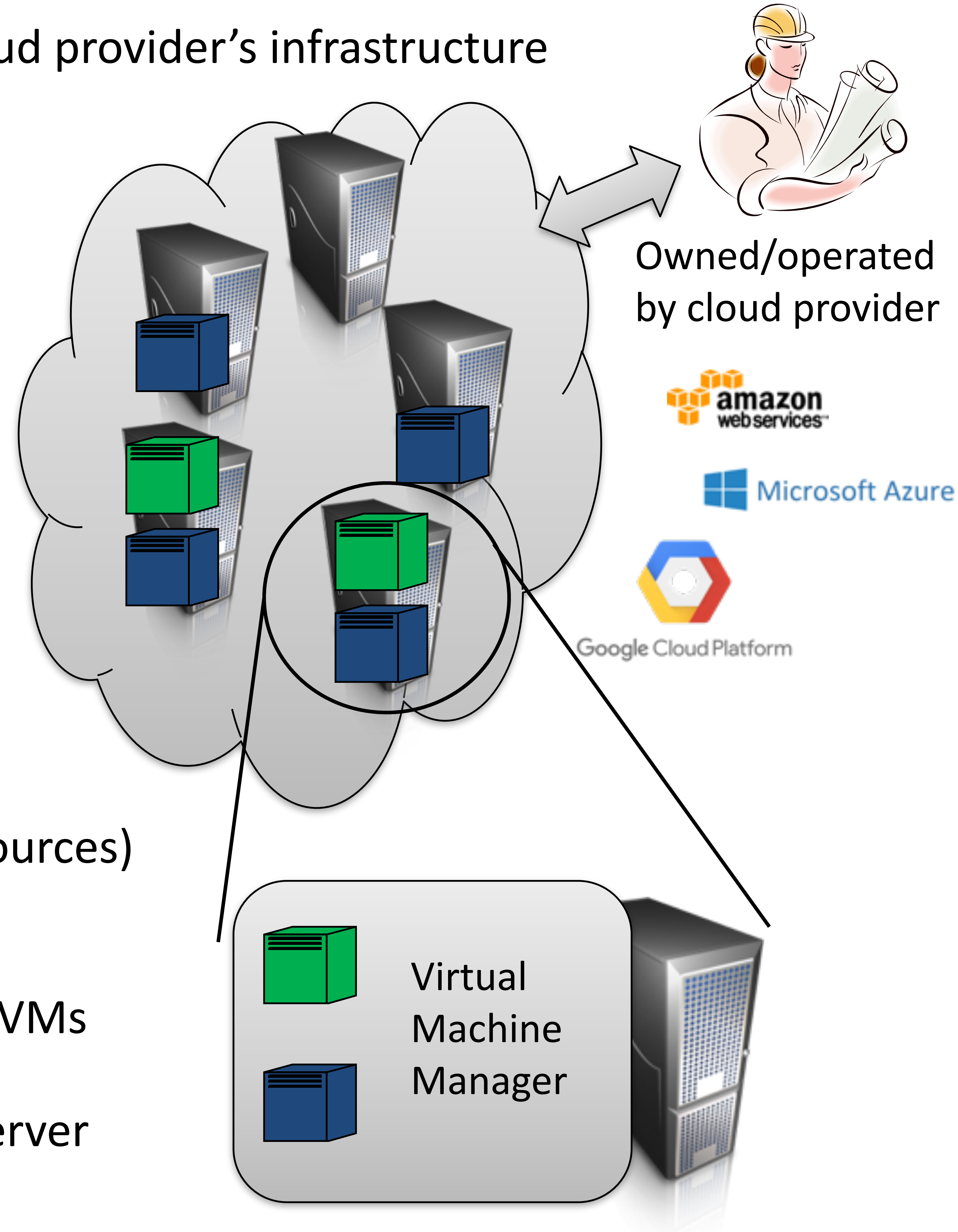


Cloud Services

A simplified model of public cloud computing

Users run Virtual Machines (VMs) on cloud provider's infrastructure



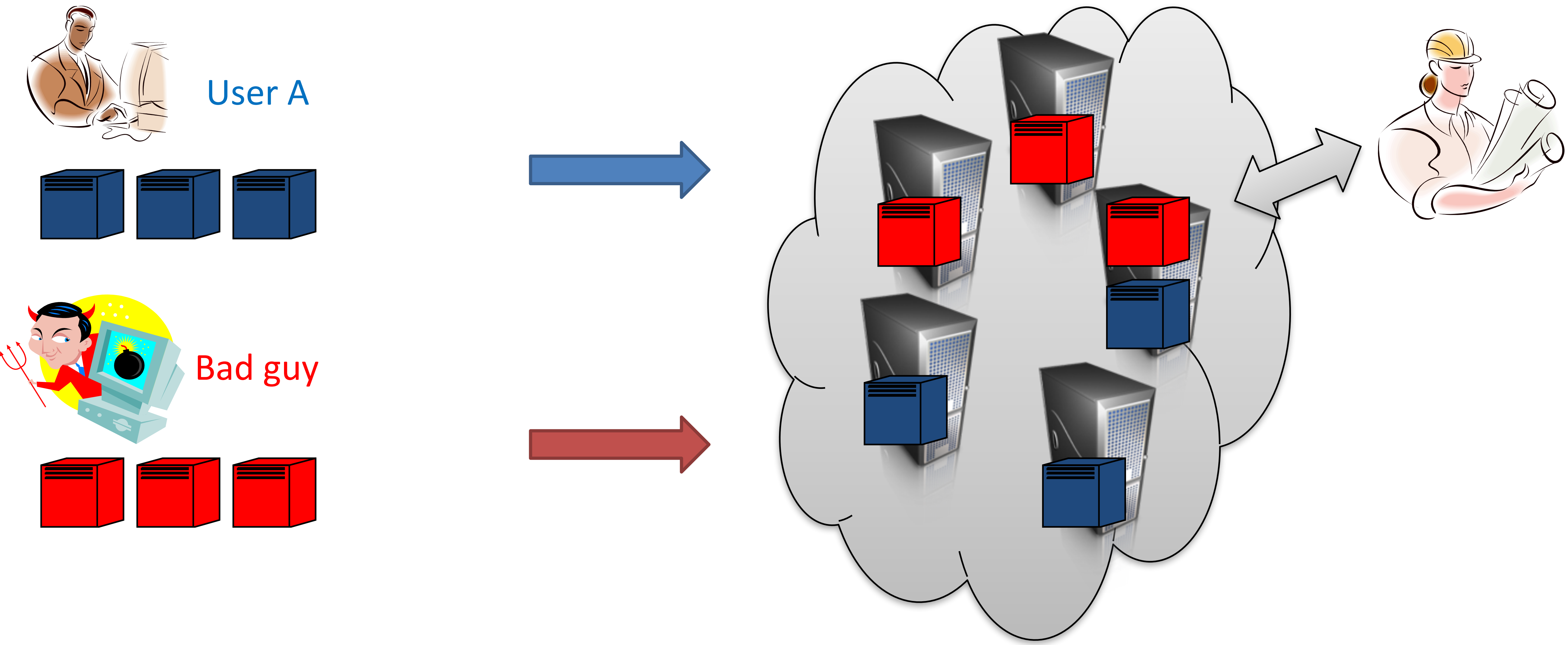


Multitenancy (users share physical resources)

Virtual Machine Manager (VMM)
manages physical server resources for VMs

To the VM should look like dedicated server

A new threat model:



Attacker identifies one or more victims VMs in cloud

1) Achieve advantageous placement via launching of VM instances

2) Launch attacks using physical proximity

Exploit VMM vulnerability   DoS   Side-channel attack

Anatomy of attack

Checking for co-residence

check that VM is on same server as target

- network-based co-residence checks
- efficacy confirmed by covert channels

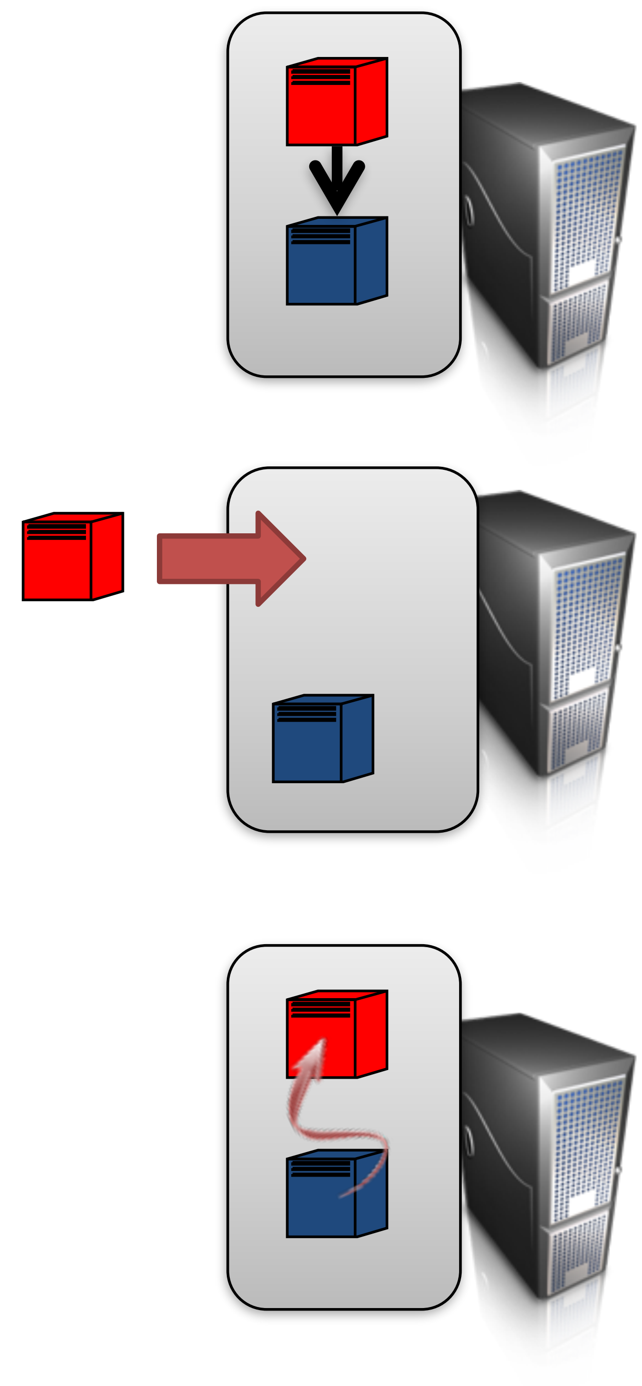
Achieving co-residence

brute forcing placement

instance flooding after target launches

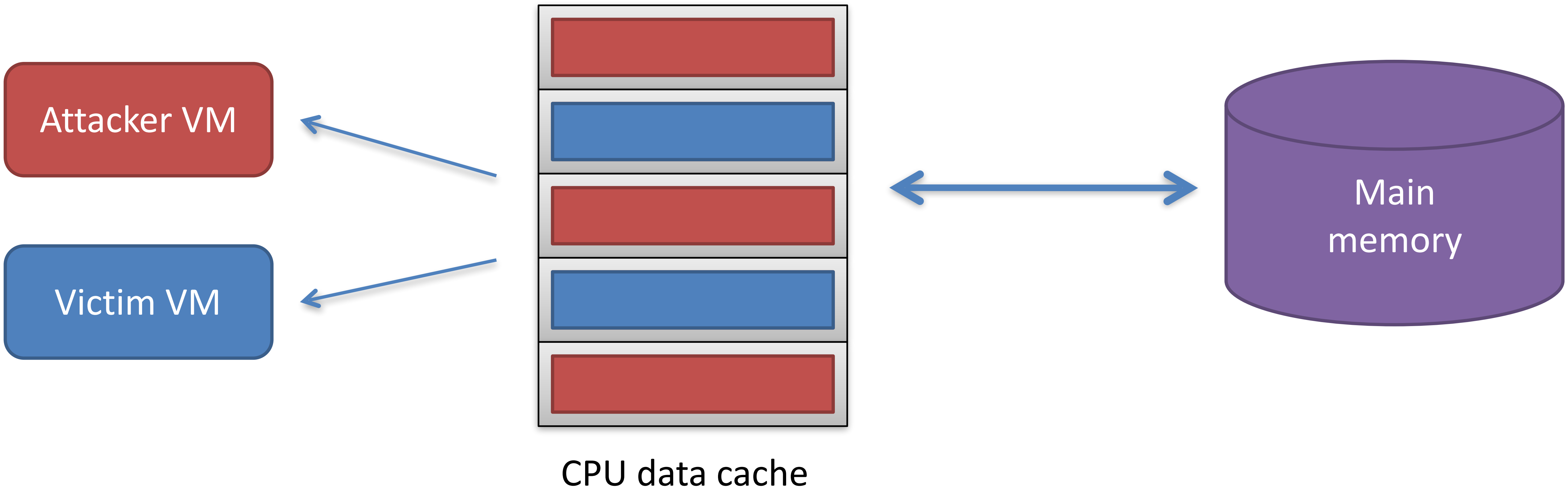
Location-based attacks

side-channels, DoS, escape-from-VM



Placement vulnerability: attackers can knowingly achieve co-residence with target

Cross-VM side channels using CPU cache contention

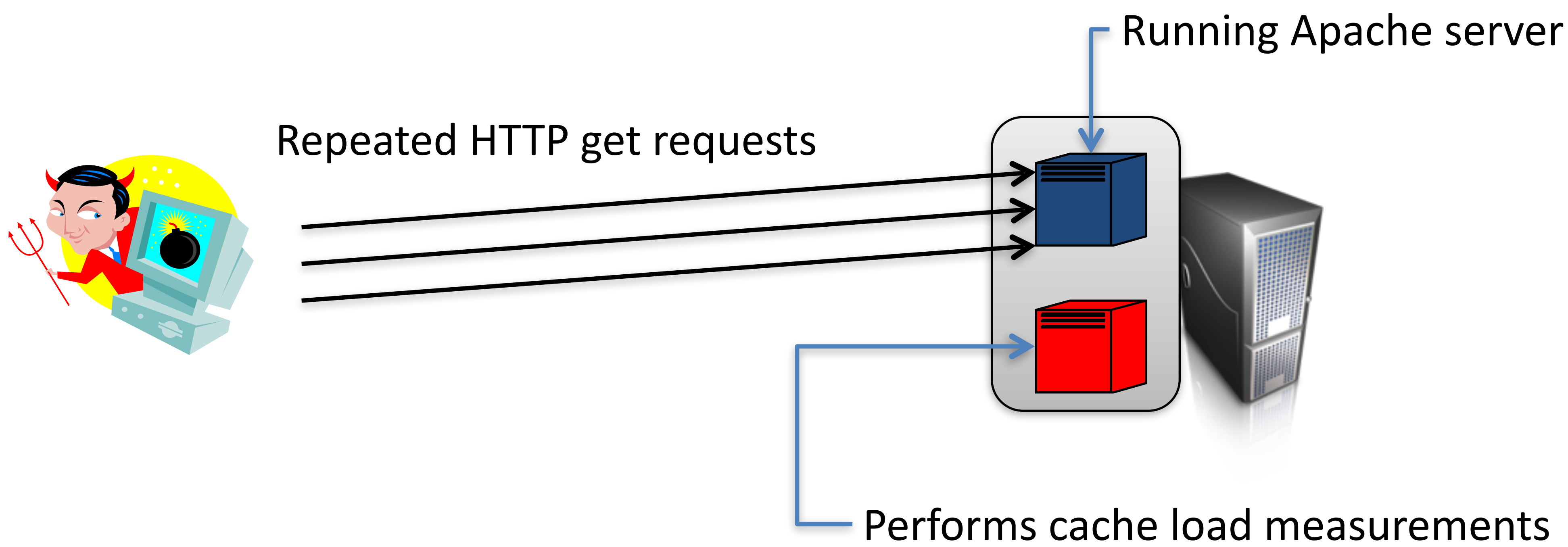


1) Read in a large array (fill CPU cache with attacker data)

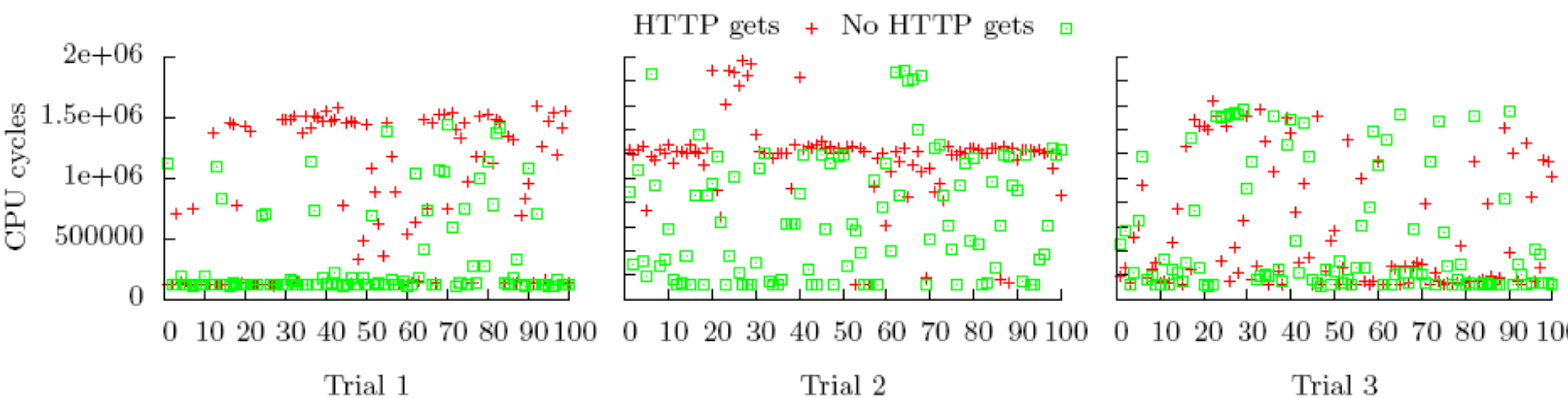
2) Busy loop (allow victim to run)

3) Measure time to read large array (the load measurement)

Cache-based cross-VM load measurement on EC2



3 pairs of instances, 2 pairs co-resident and 1 not
100 cache load measurements during **HTTP gets** (1024 byte page) and with **no HTTP gets**



Instances co-resident | Instances co-resident | Instances NOT co-resident

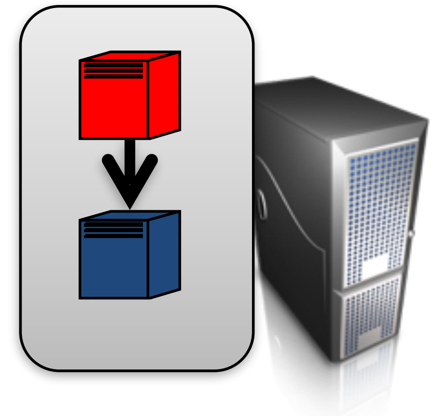
[*Hey, You, Get Off of my Cloud*, 2009, Ristenpart, et al.]

Anatomy of attack

Checking for co-residence

check that VM is on same server as target

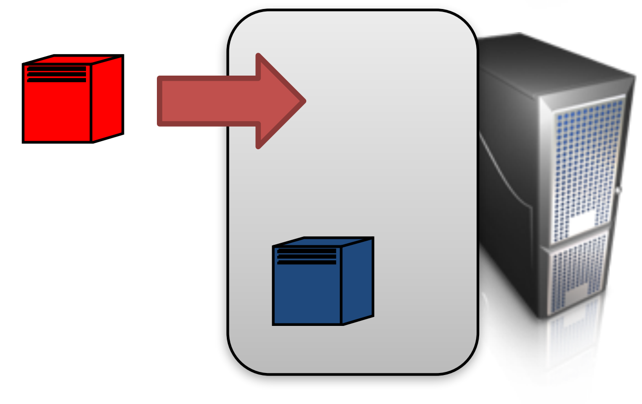
- network-based co-residence checks
- efficacy confirmed by covert channels



Achieving co-residence

brute forcing placement

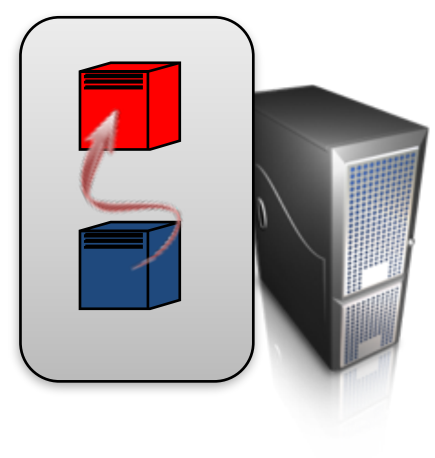
instance flooding after target launches



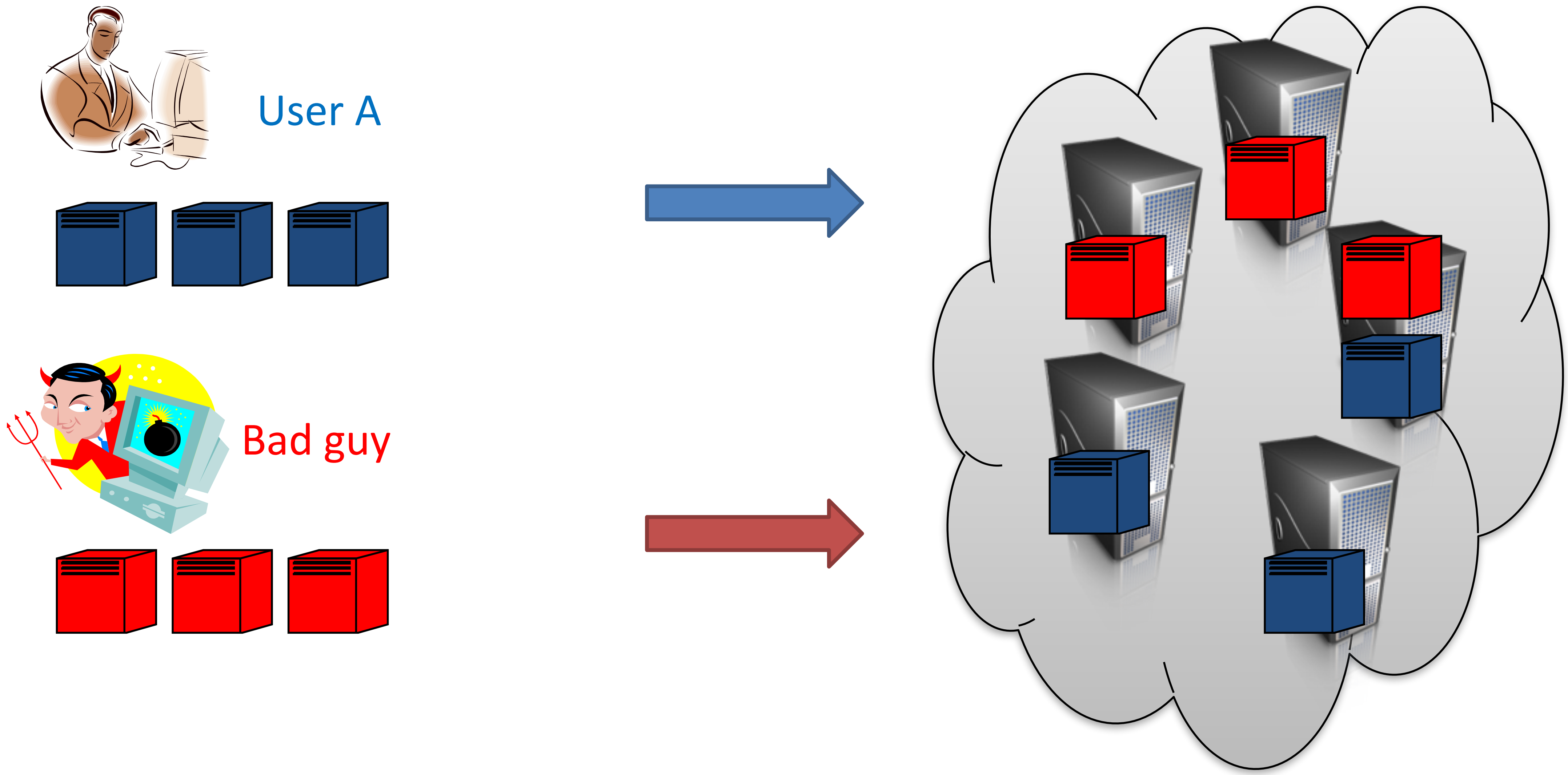
Placement vulnerability: attackers can knowingly achieve co-residence with target

Location-based attacks

side-channels, DoS, escape-from-VM



How hard should co-location be?



- **Random placement policy**
- N = 50k machines
- v=#victim VMs, a=#attacker VMs
- Probability of collision:

  $P_c = 1-(1-v/N)^a$

v	$a = \ln(1-P_c)/\ln(1-v/N)$; $P_c = 0.5$
10	3466
20	1733
30	1155

Co-location Strategies

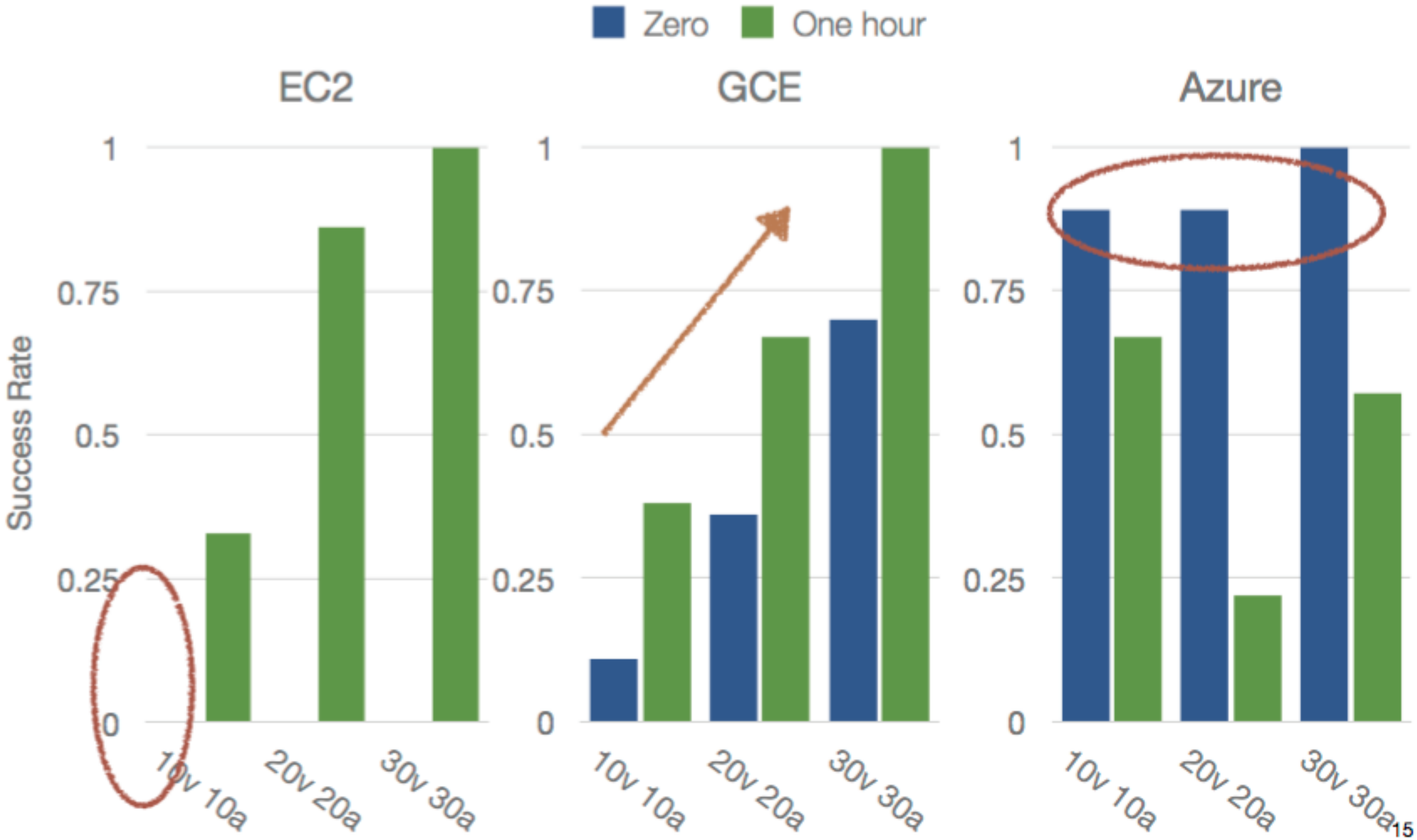
- **Basic strategy**
- Trigger launch of victim VMs
    - Drive HTTP traffic and trigger autoscaling to launch more victim VMs
- Time launch of attacker VMs in co-ordination

- How effective is this?
- How much does this cost?
- How long does this take?

Example strategies on EC2

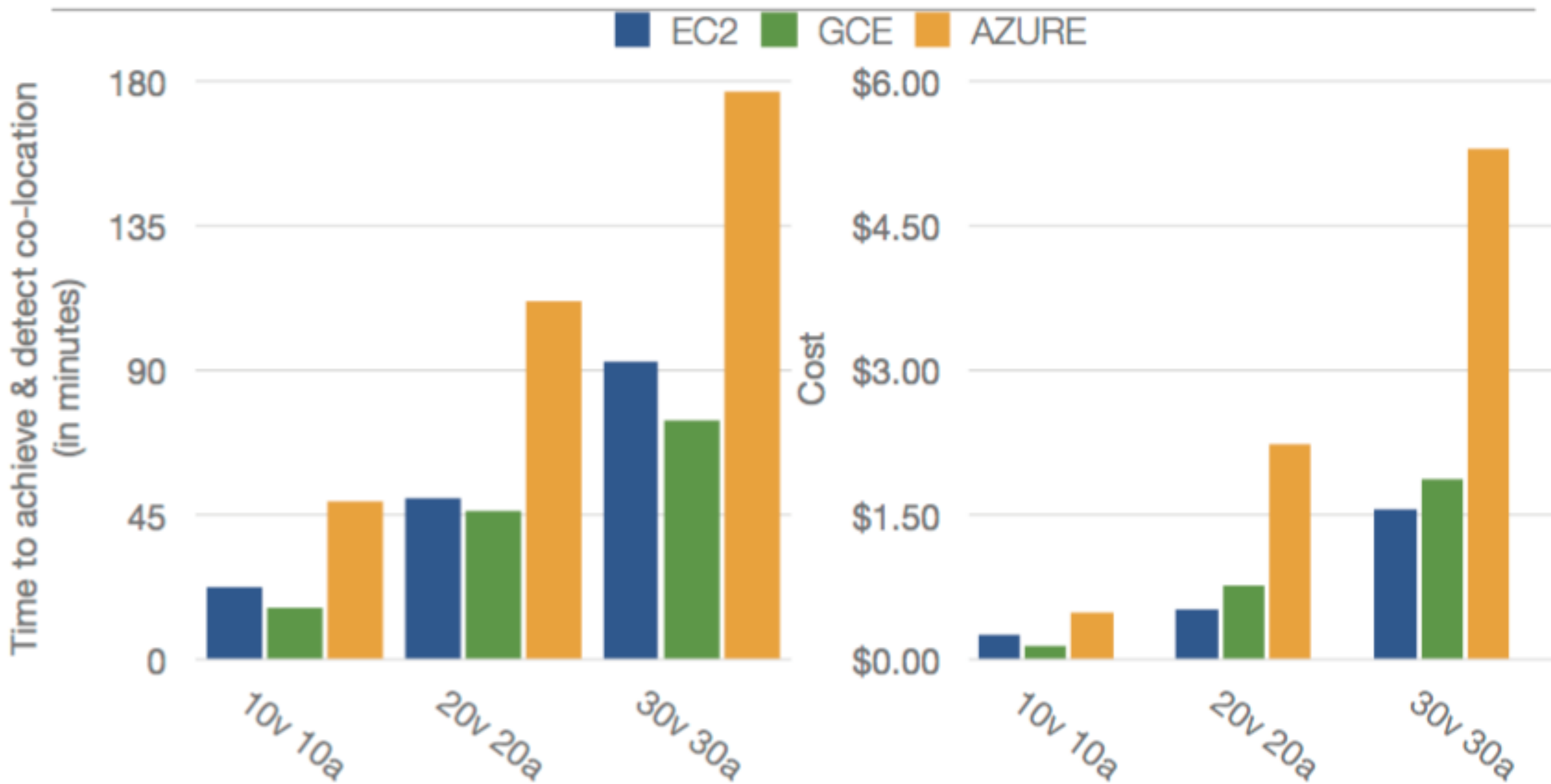
Launch Strategy	v x a
Launch 10 VMs in less popular datacenter	10x10
Launch 30 VMs 1 hour after victim VM launches	30x30
Launch more than 20 VMs 4 hours after victim VM launches	20x20

Results: Varying Delay between Launches



[*A Placement Vulnerability Study in Public Clouds*, 2015, V Varadarajan]

Cost of a Launch Strategy



- Cheapest strategy: $0.14 (GCE)
- Most expensive strategy: $5.30 (Azure)

[*A Placement Vulnerability Study in Public Clouds*, 2015, V Varadarajan] *ecrime*

Botnets

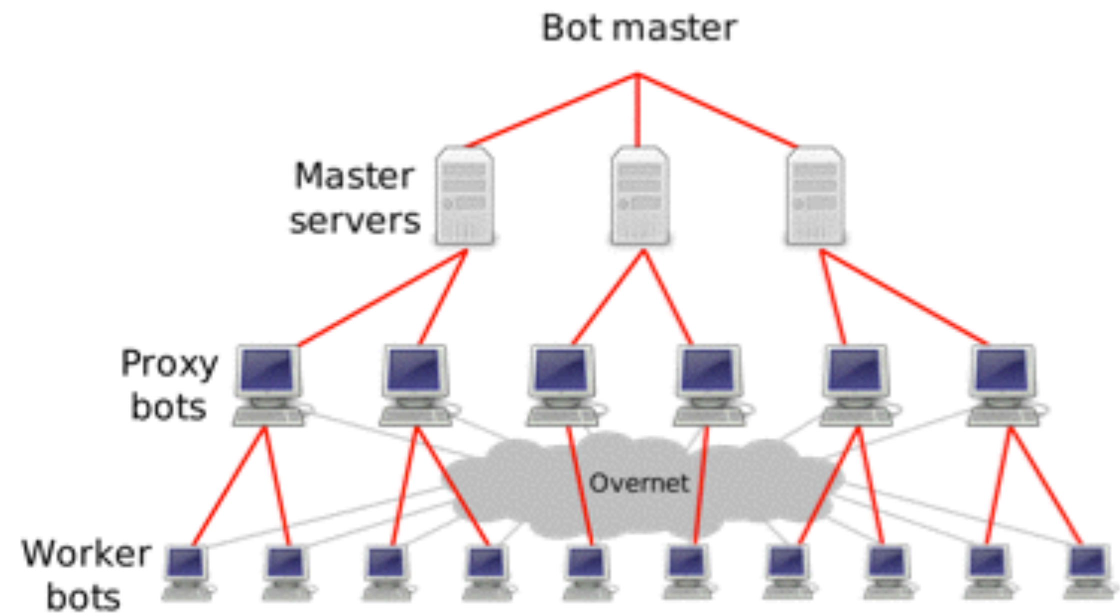


Figure 1: The Storm botnet hierarchy.

- Botnets:
  - Command and Control (C&C)
  - Zombie hosts (bots)
- C&C type:
  - centralized, peer-to-peer
- Infection vector:
  - spam, scanning, worm (self-propagating virus)
- Usage: ?

How to make money off a botnet?

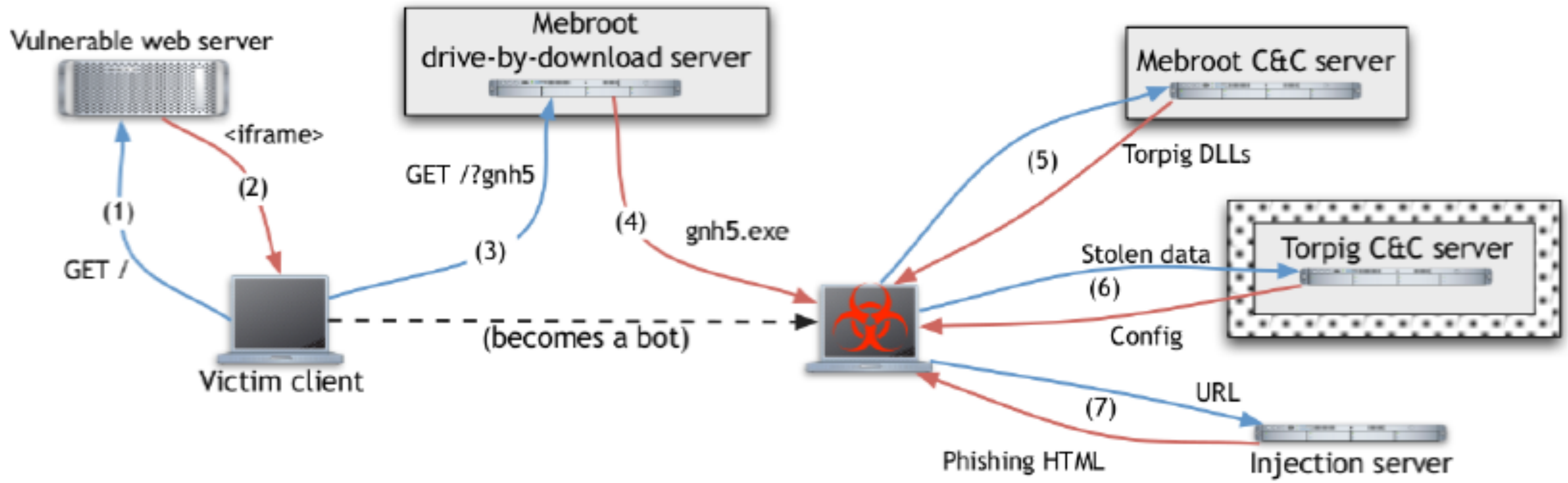
think-*pair*-share

- Rental
  - "Pay me money, and I'll let you use my botnet… no questions asked"
- DDoS extortion
  - "Pay me or I take your legitimate business off web"
- Bulk traffic selling
  - "Pay me to direct bots to websites to boost visit counts"
- Click fraud, SEO
  - "Simulate clicks on advertised links to generate revenue"
  - Cloaking, link farms, etc.
- Theft of monetizable information (eg., financial accounts)
- Ransomware
  - "I've encrypted your harddrive, now pay me money to unencrypt it"
- Advertise products

Torpig Botnet

- 2005-2009?
- 50k-180k bots
- 2008: "Most advanced piece of crimeware ever built"
- Use *domain flux* to contact command and control (C&C) servers
- Hijacked by UC Santa Barbara researchers and studied for 10 days

[*Your Botnet is My Botnet: Analysis of a Botnet Takeover*, 2009, Stone-Gross et al.]



How to join a Torpig botnet

1: Click on dodgy link to vulnerable website

2-4: Download Mebroot malware

5: Mebroot downloads Torpig DLL (your a bot!)

6: Upload all you sensitive data to Torpig C&C

7: Profit! (not yours)

Domain Flux

- Each bot generates candidate domain names for C&C servers
- Probe each one, use the first one that talks the C&C protocol
- Researchers ran the algorithm forward several weeks
- Discovered un-registered domains and registered them
- Setup their own C&C server
- Your botnet is my botnet

```
suffix = ["anj", "ebf", "arm", "pra", "aym", "unj",
    "ulj", "uag", "esp", "kot", "onv", "edc"]

def generate_daily_domain():
    t = GetLocalTime()
    p = 8
    return generate_domain(t, p)

def scramble_date(t, p):
    return (((t.month ^ t.day) + t.day) * p) +
        t.day + t.year

def generate_domain(t, p):
    if t.year < 2007:
        t.year = 2007
    s = scramble_date(t, p)
    c1 = (((t.year >> 2) & 0x3fc0) + s) % 25 + 'a'
    c2 = (t.month + s) % 10 + 'a'
    c3 = ((t.year & 0xff) + s) % 25 + 'a'
    if t.day * 2 < '0' ||  t.day * 2 > '9':
        c4 = (t.day * 2) % 25 + 'a'
    else:
        c4 = t.day % 10 + '1'
    return c1 + 'h' + c2 + c3 + 'x' + c4 +
        suffix[t.month - 1]
```

Listing 1: Torpig daily domain generation algorithm.

Stealing a botnet

- Researchers bought two domains and hosting
- Put up C&C server to capture all reported information by bots
- Controlled Torpig botnet for 10 days
- Captured 70 GBs of stolen information
- Used these data to study how big the botnet was and what it did (crime)

Estimating botnet size

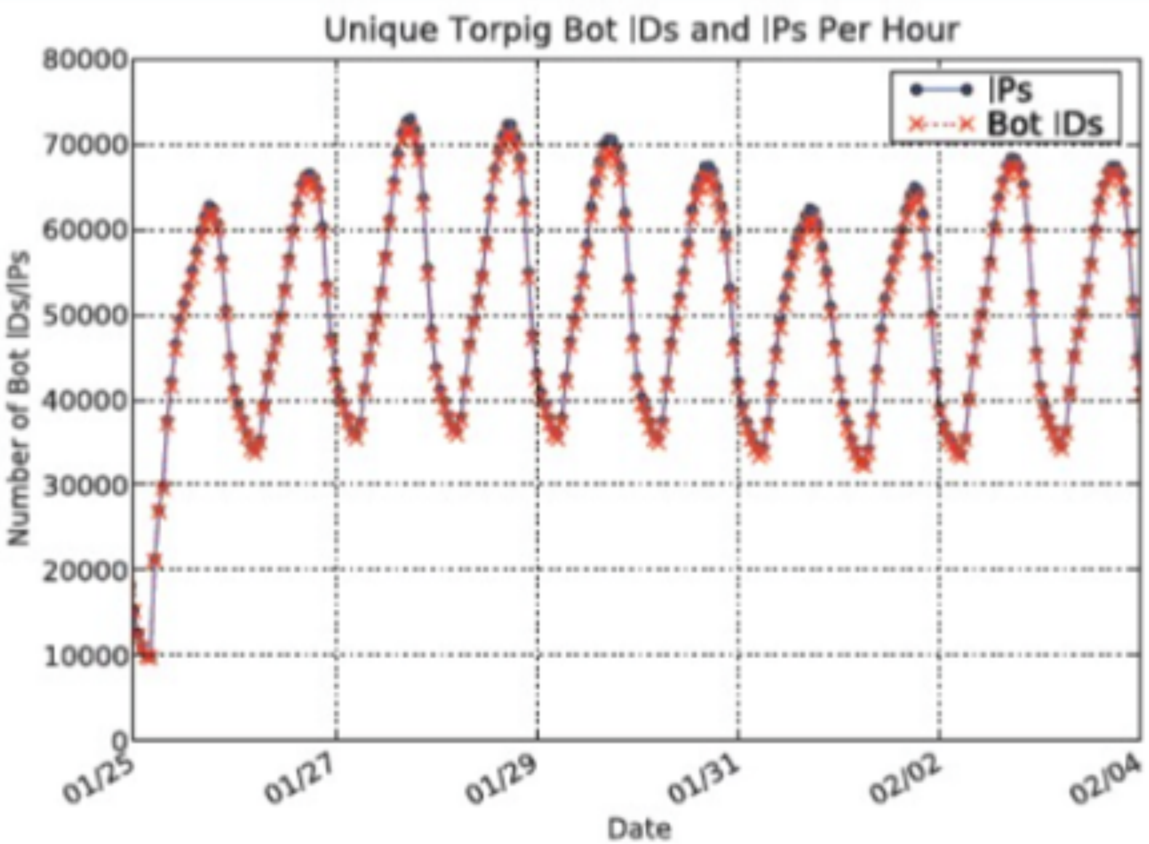


Figure 9: Unique Bot IDs and IP addresses per hour.

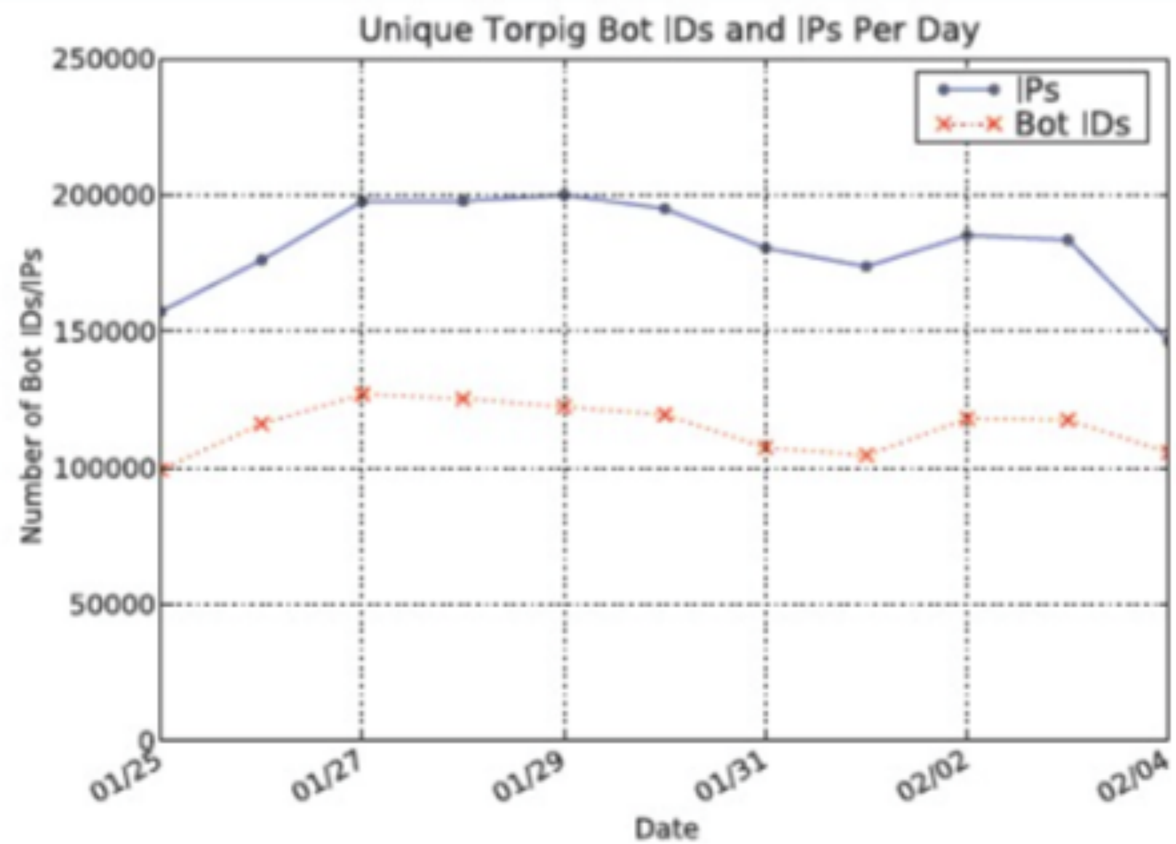


Figure 10: Unique Bot IDs and IP addresses per day.

Torpig bots report to C&C servers using a unique botnet ID
Useful for correctly estimating size

Table 1. Data items sent to our C&C server by Torpig bots.

Data type	Data items
Form data	11,966,532
Email	1,258,862
Windows password	1,235,122
POP account	415,206
HTTP account	411,039
SMTP account	100,472
Mailbox account	54,090
FTP account	12,307

Stealing Financial Accounts

In 10 days, stolen accounts from:

- Paypal (1770)
- Poste Italiane (765)
- Capital One (314)
- E*Trade (304)
- Chase (217)

Country	Institutions (#)	Accounts (#)
US	60	4,287
IT	34	1,459
DE	122	641
ES	18	228
PL	14	102
Other	162	1,593
Total	410	8,310

Table 3: Accounts at financial institutions stolen by Torpig.

Ethics

Two principles to protect victims

- PRINCIPLE 1.
  - The sinkholed botnet should be operated so that any harm and/or damage to victims and targets of attacks would be minimized.
- PRINCIPLE 2.
  - The sinkholed botnet should collect enough information to enable notification and remediation of affected parties.

recap

* Cloud computing
  / Placement vulnerabilities
  / Co-residency detection via side-channels
  / Co-location strategies

* Malware + botnets
  / Botnet uses
  / Architecture
  / Domain flux, C&C hijacking