CreepyDOL: Cheap, Distributed Stalking

Brendan O'Connor
Malice Afterthought, Inc.

So, there are three takeaways from my talk: (next slide)

Everything leaks too much data.

At every level, we've forgotten that privacy, not just security, should be a goal.

It is no longer possible to "blend in to the crowd."

Certain assumptions, and many action movies, will have to be adjusted.

Every scene where an action hero dives into a mall with 10K people and the Feds say "dang, we lost him?" Yeah, that won't work anymore.

Fundamental changes are needed to fix this.

So we're probably doomed. But it's going to be a fun time in the interim.

And I mean both technical changes---more on this later---and cultural ones: it needs to *NOT* be OK to request too much data, let alone to store it or transmit it.

Digression 1: Weev

Or Andrew Auernheimer, if you prefer.

The United States Government has declared a holy war against legitimate security research.

Some of us think that's not a good idea.

It doesn't matter whether you like Weev or not.

Mighty Casey got three strikes, but we get only one; "They claimed it was for the sake of their grandparents and grandchildren, but it was of course for the sake of their grandparent's grandchildren, and their grandchildren's grandparents." (Douglas Adams)

The time to fight private ex post facto laws is now---because once ratified by a Court of Appeals, it will be a generation before we get to try again. So set aside any dislike you may have for Weev---perhaps for the best of reasons---and act in your own enlightened self-interest. Or everyone in this room will be in prison soon.

Amicus Brief of Meredith Patterson, Brendan O'Connor, Sergey Bratus, Gabriella Coleman, Peyton Engel, Matthew Green, Dan Hirsch, Dan Kaminsky, Samuel Liles, Shane MacDougall, Jericho, Space Rogue, and Mudge

And Alex Muentz, another hacker and a full lawyer, who was willing to take a law student's brief and submit it to the Circuit Court of Appeals.

All of the names on this list are big deals. Meredith Patterson from LangSec, Sergey Bratus, Patron Saint of the Gospel of Weird Machines, Crypto Engineer and Professor Matt Green, Dan Kaminsky, Jericho, Space Rogue, Mudge... the list goes on. And that should tell you how scared the entire community is, and should be; it touches all of us, whether we're DARPA program managers, professors, or itinerant hackers.

In the meantime, there will be a chilling effect, as we cannot trust legal actions not to be prosecuted anyway.

Therefore, CreepyDOL has not been used to take on an entire city. It's been tested, and parts of it have been tested with extremely high amounts of data, but I leave the next step, world domination, to a braver researcher.

Extremely Serious Disclaimer

This presentation does not create an attorney-client relationship. Probably. If it does, it will have said it does. Although it could have created an attorney-client relationship without explicitly saying so, because the law is tricky like that.

This presentation may contain confidential and/or legally privileged information. If it does, and you are not the intended recipient, then the sender hereby requests that you notify him of his mistake and destroy all copies in your possession. The sender also concedes that he is very, very stupid.

This disclaimer is not especially concerned with intelligibility. This disclaimer has no qualms about indulging in the more obnoxious trademarks of legalese, including but not limited to (i) the phrase "including but not limited to", (ii) the use of "said" as an adjective, (iii) re-naming conventions that have little to no basis in vernacular English and, regardless, never actually recur (hereinafter referred to as "the 1980 Atlanta Falcons"), and (iv) lowercase Roman numerals.

This disclaimer exists for precisely one reason—to make this presentation appear more professional. This disclaimer shall not be construed as a guarantee of actual professionalism on the part of the sender. Any actual professionalism contained herein is purely coincidental and is in no way attributable to the presence of this disclaimer. If you aren't reading this, then this disclaimer has done its job. Its sad, pointless job. THIS DISCLAIMER IS NOT INTENDED TO BE IRONIC.

Adapted, with kind permission from the author and publisher, from http://www.mcsweeneys.net/articles/alright-fine-ill-add-a-disclaimer-to-my-emails .

DARPA Cyber Fast Track

- CreepyDOL is not CFT work
  - DARPA tries hard not to build stuff that creeps people out this much, and they're very nice people.
  - That said, two CFT contracts did let me build two of the core systems: Reticle, and the visualization system.
  - Thanks, Mudge!

Roadmap

- Goals
- Background
- Architecture
- Design of CreepyDOL
- Future Work
- Mitigation

Goals

- How much data can be extracted from PASSIVE wireless monitoring?
- Well, rather a lot, really, but how much can we do for really, really cheap?

I. Goals

A. How much data can be extracted from passive wireless monitoring?

1. More than just from a network trace---remember that when not connected to a wireless network, WiFi devices send out lists of their known networks, asking if anyone can help them.

2. As soon as a device thinks it's connected to WiFi, all its background sync services will kick off again---DropBox, iMessage, all the rest. So we'll immediately know that certain services will be in play.

3. Over unencrypted WiFi, all the traffic sent by a device is exposed. Even if we can't see both sides of every message, we can learn a lot from what we do see---especially if we know how a given protocol operates.

4. How much better could we do if we had not one sensor, but ten? Spread out over an area? Now we have geolocation, time and place analysis, etc.

5. If we're tracking over a large area, we don't just want to know traffic and devices: we want to know people. Can we take data and find people? (I don't want your SSN, I want your name. And really, I want to know enough about you to blackmail you; information is control.)

Goals

- Can we do large-scale sensor networks without centralized communications?
- This makes it cheaper, faster to deploy, easier to use, and much more scalable...
- It's also much harder to attack.

Friday, August 2, 13

B. Can we do large-scale sensing without centralized communications?

1. If we centralize communications, life is simple; everyone phones home---but a compromised node gives every attacker the location of the mothership.

2. Centralized communications decrease resistance to attack, and prevent you from responding agilely to attack.

Goals

- Can we present massive amounts of data in a way that doesn't make people's brains hurt?
- Hint: the PRISM slides make Tufte cry

Friday, August 2, 13

C. Can we present massive amounts of this data in a way that is intelligible by mortals? User-friendly? Still secure?

1. Group One of high security products: incredible technology, terrible UI. This causes low adoption, or (possibly worse) mistakes in use. Systems fail, people die. Examples: Pidgin-OTR, or PGP/OpenPGP.

2. Group Two: Concerns about technology, great UI. This causes adoption, but can cause massive problems later (if the concerns are borne out). Examples: HushMail, or the Silent Circle ZRTP issues.

3. Group Three: Good technology, great UI. This is wonderful, but incredibly hard to do (because UI masters are usually not security wizards). Example: CryptoCat, RedPhone.

4. We would aspire to have CreepyDOL, and especially the underlying Reticle communications technology, be in Group Three, through a variety of methods to ensure secure communication in relatively-intelligible ways. *This is an ongoing process.* Our code is open source, to allow verification, and will be released in the coming weeks.

Roadmap

- Goals
- Background
- Architecture
- Design of CreepyDOL
- Future Work
- Mitigation

Background: Sensor Networks

- Academic researchers *rock* at this!
  - MANETs
  - Great sensors, very sensitive
  - Extremely (extremely!) low power
- Unfortunately, the cost is severe: can be several hundred $ per node
  - Poor grad student, and law school won't pay for CS research! So we need something different for hardware.

II. Background

A. Sensor Networks

1. Academic researchers have spent tons of time and resources on these. MANETs, other advances in technology have resulted.

2. A lot of these have uW power levels, and sacrifice languages, OS, and cost to get there---especially cost, with many nodes costing $500 or more. Each.

3. I can't afford this. I want something I can afford to break, to lose, and even to have stolen. I want it an order of magnitude cheaper, and I want it to run Linux. (Ubuntu or Debian, if possible.)

Background: Large-Scale Surveillance

- In my original outline, submitted in March: “One can assume that they [the IC] have solved all of the problems involved in CreepyDOL before me, and that they should, rightfully, be cited as prior art. I'd love to do so; as soon as they publish their work, I'll be happy to cite them.”
- Heh... heh.
- Pour one out for the Intelligence Community: a lot of this stuff is a pain to figure out

B. Large-Scale Surveillance

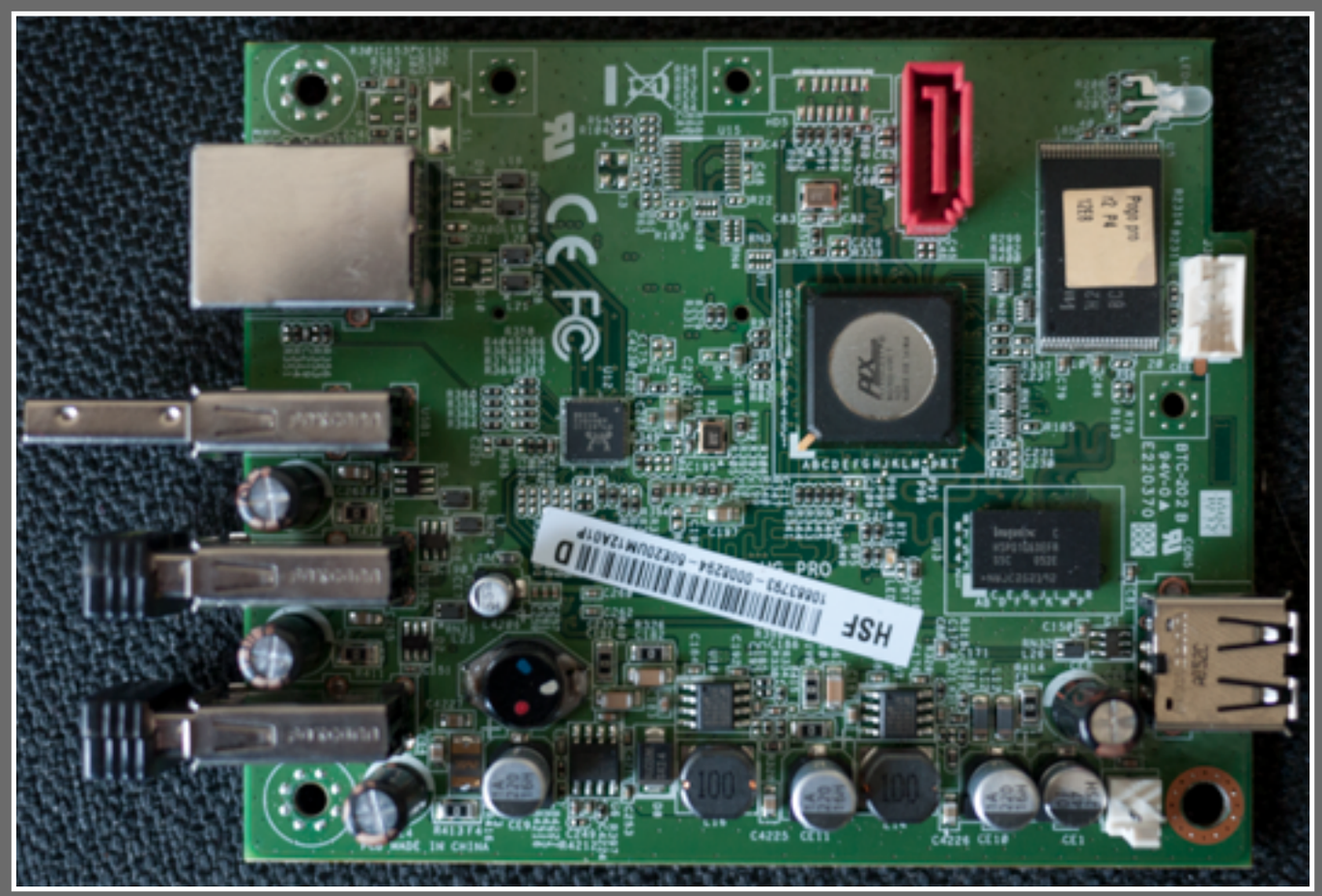
1. It's commonly believed that the US Government has the ability to monitor all network traffic in the US. This is not helped by the fact that they've actually said that in the last few months.

2. One can assume that they have solved all of the problems involved in CreepyDOL before me, and that they should, rightfully, be cited as prior art. I'd love to do so; as soon as they publish their work, I'll be happy to cite them.

Roadmap

- Goals
- Background
- Architecture
- Design of CreepyDOL
- Future Work
- Mitigation

Hardware!

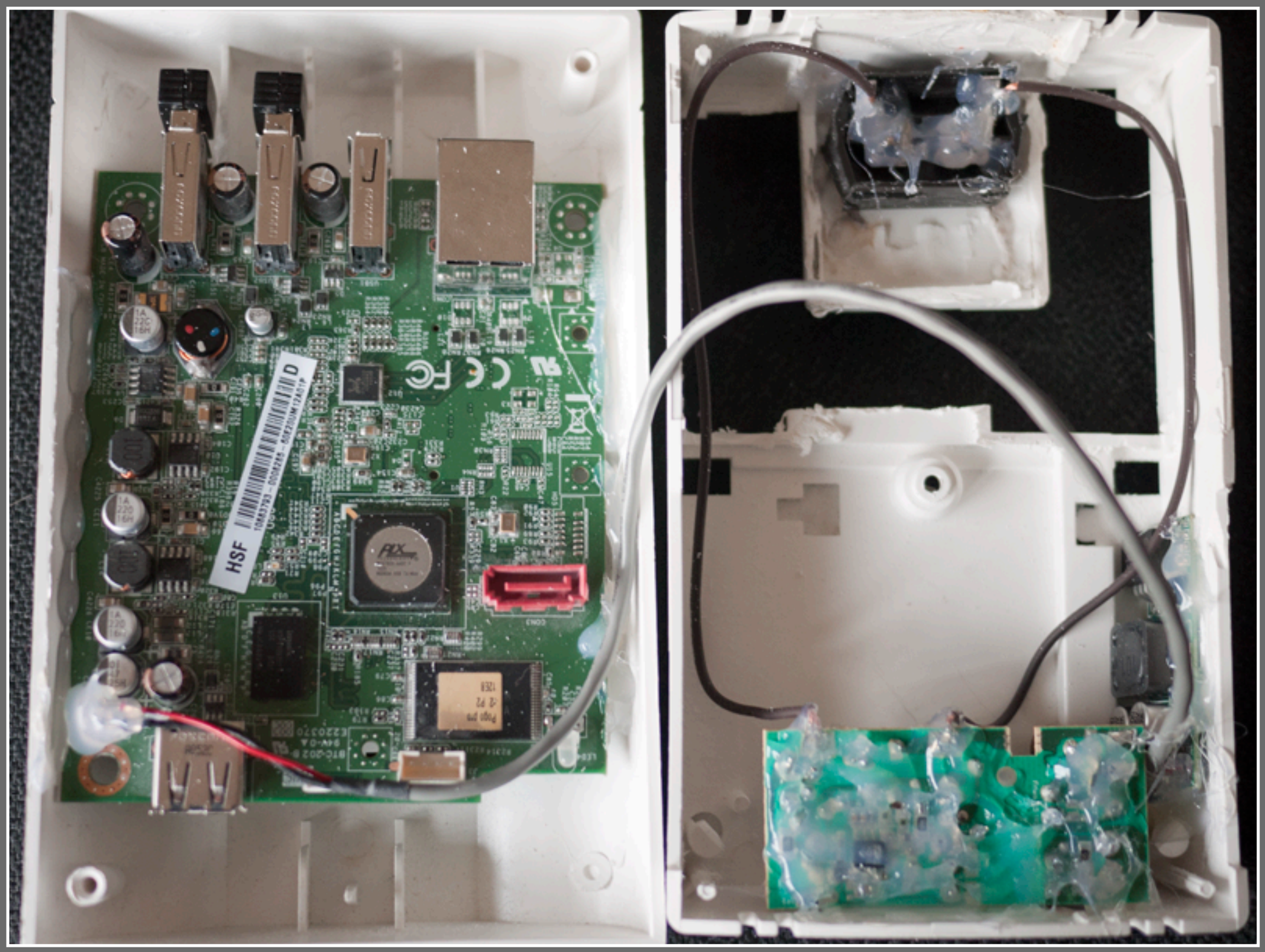


F-BOMB v.1 (ShmooCon 2012)

A. Hardware: F-BOMB, version 2 (Falling/Ballistically-launched Object that Makes Backdoors)

1. Originally presented at ShmooCon 2012. At that time, this was based on the Marvell Sheeva board, the same board used by the Pwnie Plug that's been selling so well for years. To keep costs down, I was actually buying PogoPlugs, a rebranding of the Sheeva board, as they were being sold as essentially fire sales, and stripping out their guts.

Conveniently, (next slide)



this also fits well into, just as an example, a carbon monoxide detector. How many of you have checked your CO detector to make sure it wasn’t a hidden sensor network working for me?

F-BOMB v.2

2. Now based on the Raspberry Pi Model A, because it's awesome, runs an easier version of Linux (Debian vs. Arch), and I can actually get it for cheaper than the salvage PogoPlugs. We also get significantly reduced power consumption, it runs at a better voltage (5v instead of 12v), it's physically much smaller and lighter, and it actually has more RAM and processing power on board. You can see there's a bit of cord sticking out of each F-BOMB in this photo; this is because I mis-measured when buying the cas. But the Raspberry Pi is actually much smaller than the Sheeva board, so it fits better into smaller objects. (Hold up one.)
These devices use USB power, which means that I can plug them into walls (you can see an Apple-style USB power adapter in the lower-left), but also into USB batteries, MintyBoost kits, or anything else that gives me 5v in this ubiquitous form factor. They do not use that port as a data port.

Hardware Cost

- Raspberry Pi, Model A: $25
- Case: $4.61
- USB Hub: $5.99
- WiFi: 2x $6.52
- SD Card: $6.99
- USB Power: $1.45
- Total: 57.08 per node

Friday, August 2, 13

This is the cost list: $57.08 per node, which means it's within the price range of any kid who mows lawns energetically for a few weekends to build a group of these.

Wait... why 2 WiFi?

- Because I’m cheap and lazy
- Introducing PortalSmash: it clicks on buttons, so you don’t have to

4. Nodes don't bring "phone home" communications gear, e.g., a 3G card; that's too expensive and *very* easy to trace (just call VZW tech support!). They use PortalSmash, Open Source software I've developed to look for open (or captive portal) WiFi and use that. In an urban area, that's perfectly sufficient. (No, PortalSmash doesn't look at encrypted WiFi; yes, you could add Reaver etc. No, I'm not planning to.)

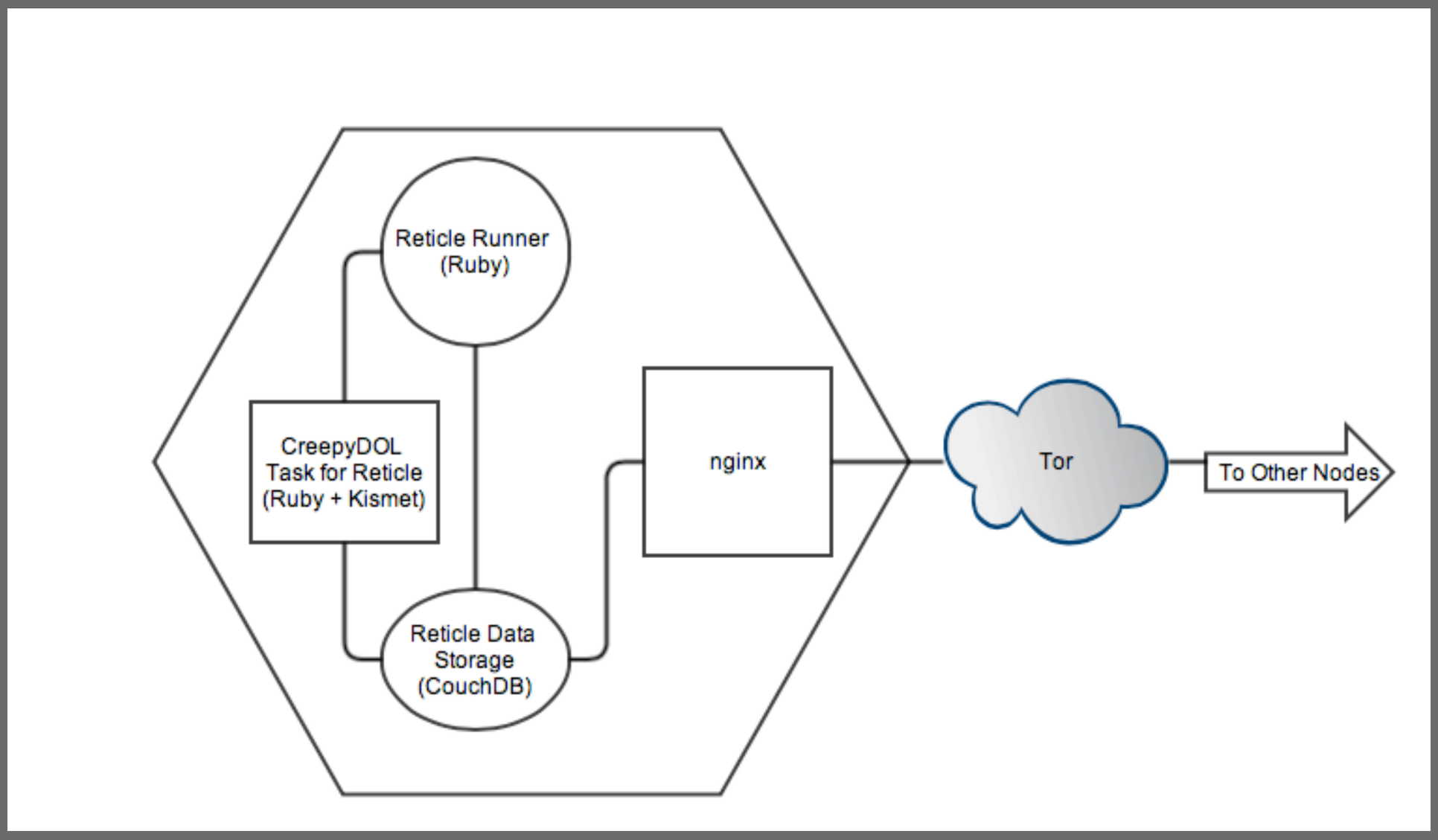
C&C Software

- "Reticle: Leaderless Command and Control"
- This was the first of the two DARPA CFT contracts I mentioned
- Whole presentation at B-Sides Vegas 2012---but I will summarize

B. C&C Software: Reticle, Leaderless C&C

1. Developed under DARPA Cyber Fast Track, Spring 2012
2. Original work presented at BSidesLV 2012, but massive improvements, and a complete rewrite, since then.

Reticle



Each Reticle node runs CouchDB, a NoSQL database, plus Nginx, Tor, and some custom management software. This lets nodes combine into a peer-to-peer “contagion” network in which each node sends commands and data to every other node, for both command infiltration and data exfiltration, without any single point of failure. They speak via Tor, to prevent anyone on the network to which they connect from determining where other Reticle nodes are living.

To make reverse-engineering of a node much more difficult, Reticle nodes can be configured with what I call “grenade” encryption: pull pin, throw toward adversary. They load their encryption keys for their local storage at boot from removable media, which is then removed to prevent an adversary from recovering the data. A “cold boot” attack is certainly possible, but since most nodes don’t have batteries, it’s physically kind of a pain to do---and it’s not a usual thing for most people to dump liquid nitrogen on the first black box they see plugged into a wall.
CreepyDOL, then, is just a mission Reticle runs; it can be retasked at any time.

Roadmap

- Goals
- Background
- Architecture
- Design of CreepyDOL
- Future Work
- Mitigation

CreepyDOL Design

- Distributed querying for distributed data
  - Since we're not bringing our own bandwidth, it would be tacky to ship a live network capture home---especially via Tor
  - So we push as much computation to the endpoints as possible

A. Distributed Querying for Distributed Data

1. Since we don't have independent, high-bandwidth channels for sending data home, it's not a good idea (and may not be possible) to send raw packets home. Nodes should send home data that's already been digested.

2. So: we run any queries on the nodes that can be effectively run on the nodes, *given data that node has collected*.

3. We do not process multi-node data on individual nodes, even though every node has access to all the data (see "contagion network"), because they've got limited processing power---and more importantly, data storage.

CreepyDOL Design

- Centralized Querying for High-Level Questions
  - This means questions that aren't answered from just one capture, like "where does he usually go for coffee in the morning?"
- Run on a backend, powered by a data sink node

B. Centralized Querying for High-Level Data

1. Things that need datapoints from multiple nodes---tracking, pattern analysis, etc., go on the "backend."

2. The backend is just another node, but with a special mission configuration: rather than just sensing and adding data, it receives data from the contagion network, pushes it into another system (a data warehouse), and then instructs the contagion to delete it to make room.

Data Query Methodology: NOM

- O: Observation
- N: Nosiness
- M: Mining

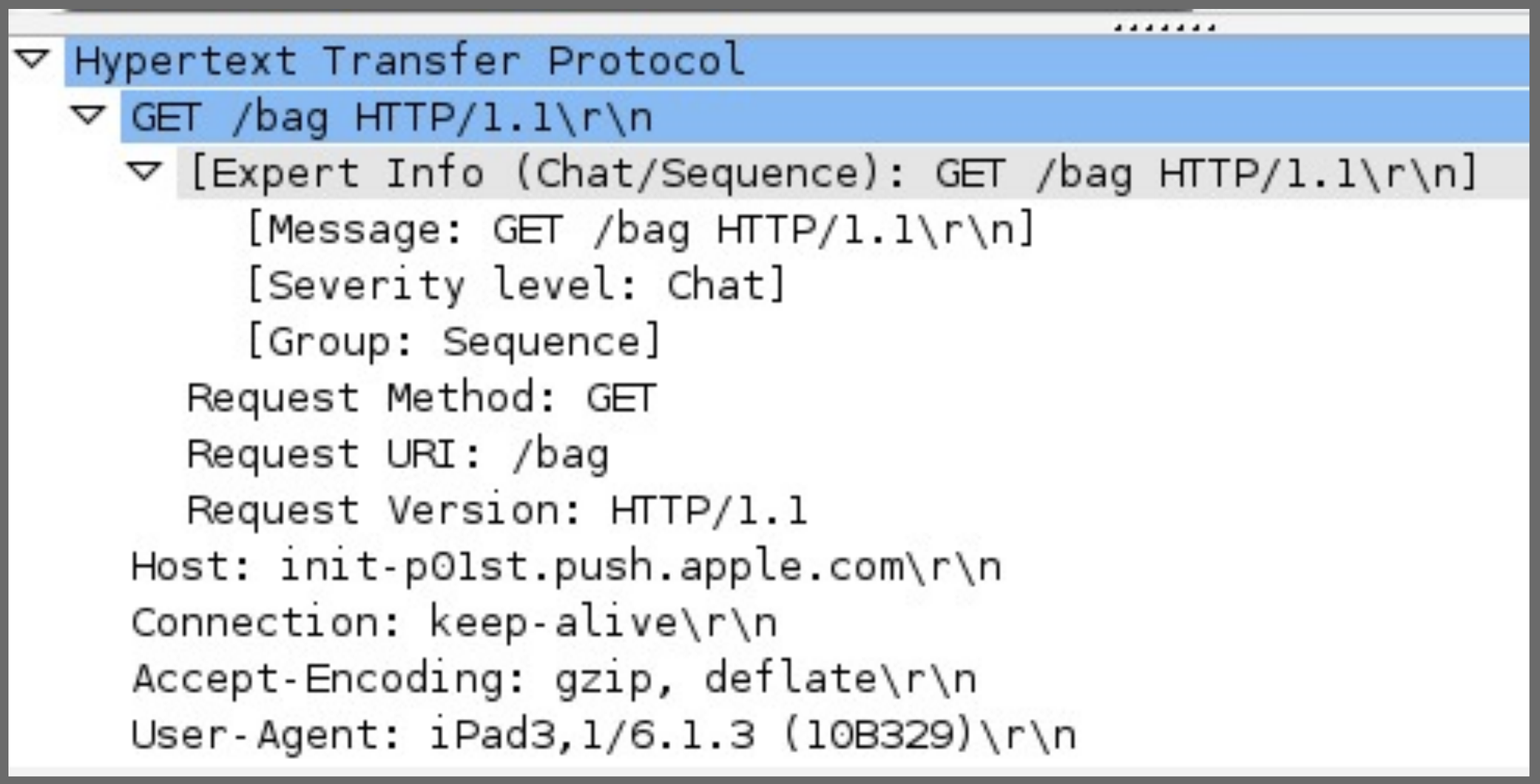
C. Data Query Methodology: NOM

1. O: Observation. Take as much data out of local traffic as possible; this means names, photos, services used, etc. To make this easy, we've created a large number of "filters" that are designed for traffic from specific applications---DropBox, Twitter, Facebook, dating websites, etc. Now, many of these services encrypt their traffic, which is admirable; however, in many cases, we can still get useful data that they provide in, e.g., their User Agent. And there's no reason for them to do this.

This is a distributed query (run on the nodes).

2. N: Nosiness. Using data extracted from O queries, there are lots of leveraged queries we can make; for instance, given an email address, we can look for accounts on web services, or given a photo, we can look for copies of that photo pointing to other accounts. This can be run either as distributed or centralized.

3. M: Mining. Taking data found by the nodes, build up larger analyzed products. For instance, is the device (person) usually in one area during a certain time of day? Are there three devices that are almost always seen together, if at all? (The latter may indicate that they are all carried by the same user.) This type of query is exclusively run on the backend.



So this is a screenshot from Wireshark, of a packet being sent to request new iMessages from Apple. Notice at the bottom, where it sends the hardware device and iOS version, as part of the HTTP header? This is unnecessary, and it's harmful. (If Apple needs this information, it could transmit it inside TLS.)

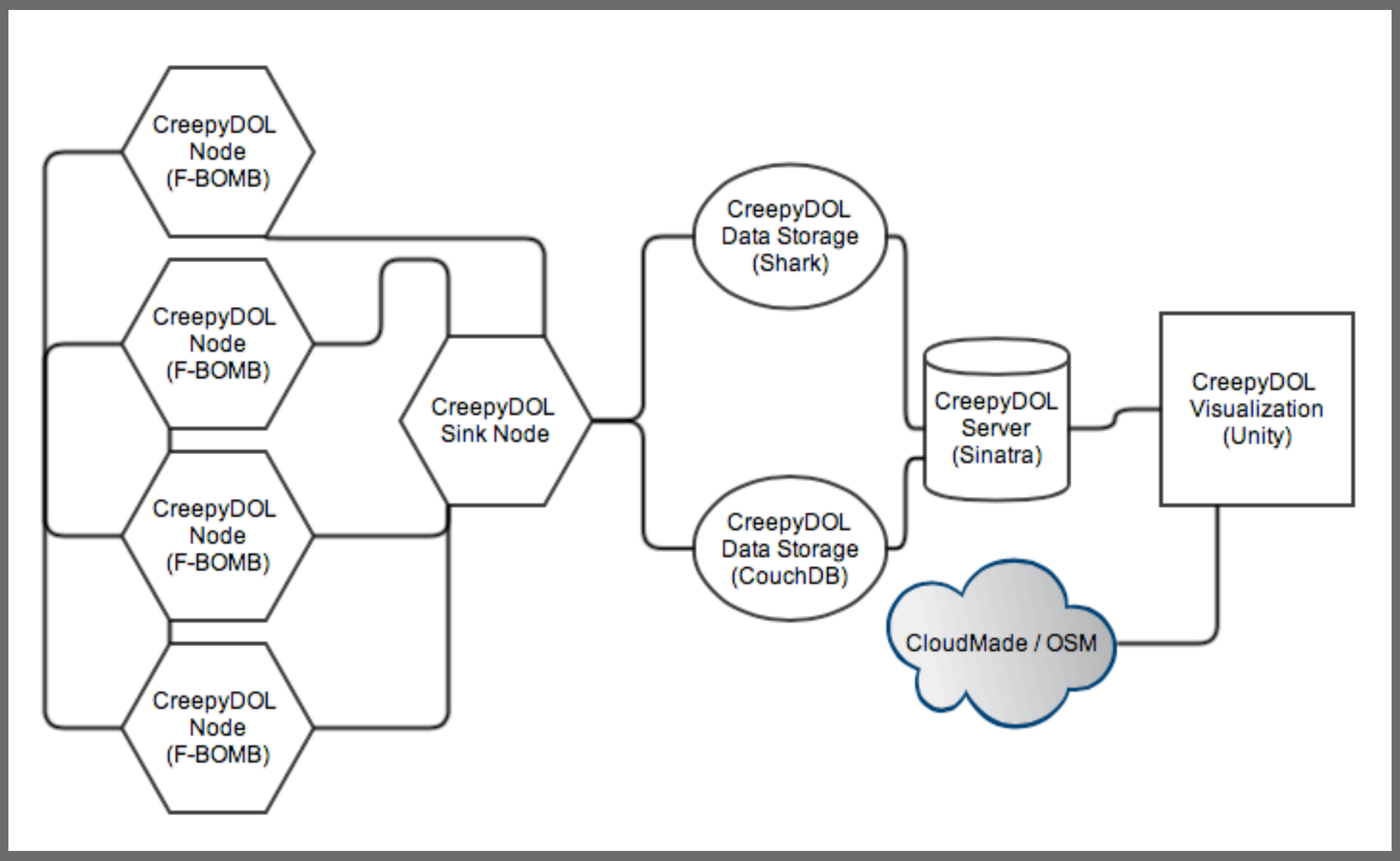
Data Query Methodology: NOM

- O: Observation
- N: Nosiness
- M: Mining

Friday, August 2, 13

2. N: Nosiness. Using data extracted from O queries, there are lots of leveraged queries we can make; for instance, given an email address, we can look for accounts on web services, or given a photo, we can look for copies of that photo pointing to other accounts. This can be run either as distributed or centralized.

3. M: Mining. Taking data found by the nodes, build up larger analyzed products. For instance, is the device (person) usually in one area during a certain time of day? Are there three devices that are almost always seen together, if at all? (The latter may indicate that they are all carried by the same user.) This type of query is exclusively run on the backend.



So this is the overall architecture for CreepyDOL. The nodes connect to each other, and one node becomes a “sink node” from which data is pulled and sent to the CreepyDOL storage, so that it can be used in the visualization. The visualization pulls data from the storage and from an OpenStreetMaps provider, to have underlaid maps.

Visualization

- Second DARPA CFT Contract
- Used the Unity Game Engine
  - Side note: wow, that's a fun toy
  - Side note: wow, I hate writing JavaScript that's interpreted by C#, then compiled into .NET CLR, then interpreted at runtime by Mono
- Runs on an iPad! Or OSX/Windows/Linux/Android
  - I think I could make it run on an XBox360, actually (Unity is Very Nice)

So let's talk about visualization.
To prevent the user (the person requesting data) from being tied to a particular computer, we use the backend to run queries for visualization, then serve the results to the user's visualization computer.
To make it easy to do large-scale visualization, I used an existing engine: the Unity game engine, used in hundreds or thousands of iPad, iPhone, XBox, Wii, and PC games. This let me take advantage of the hundreds of person-years of development they've already done to make it fast. As a side effect, it also means I can run my visualization on an iPad; since all the processing is done on a visualization server, it doesn't need to be able to hold the data in RAM.

Demo video!

But first,

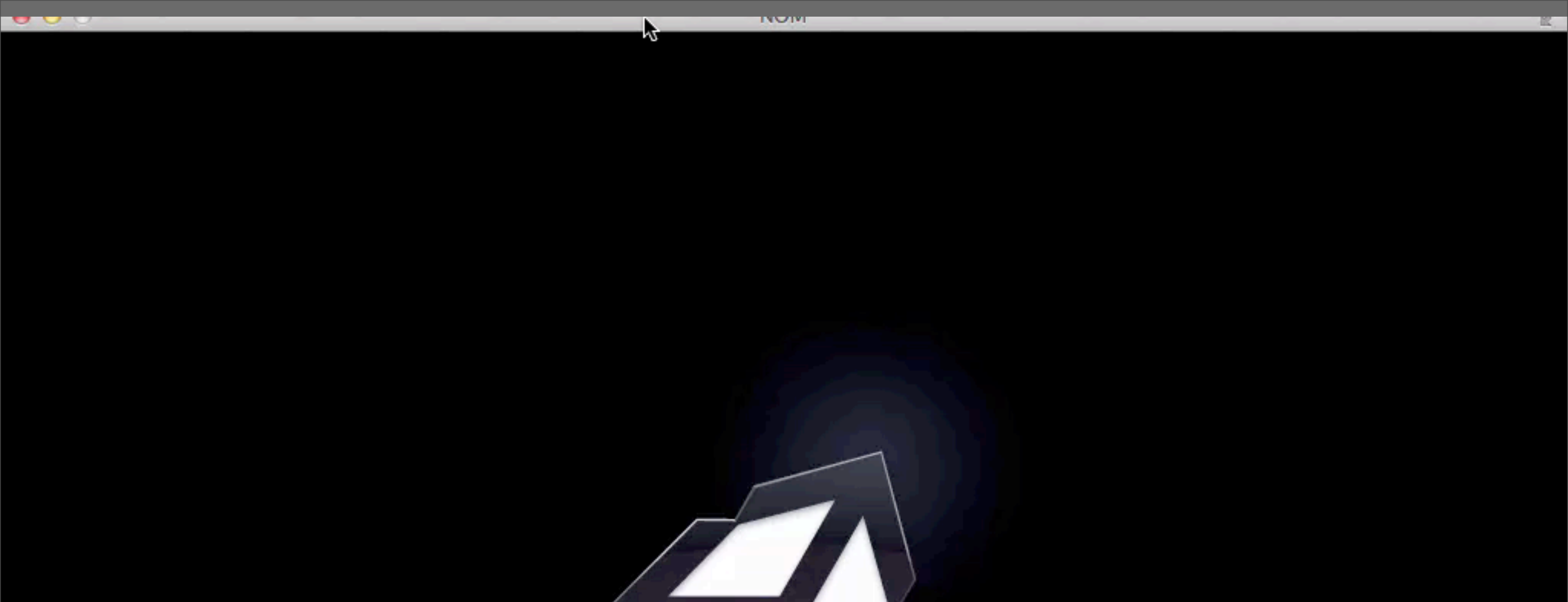
Test Parameters

- To prevent badness, we programmed the NOM system to look only for traffic from devices we owned; **no "random stranger" data was collected at any time.**

A. Test Parameters

1. Test area (map, sensor node locations)

2. ROE: No data collection on nodes that aren't in a selected set of MAC addresses that are known to us (friends). This is a terrible, unrealistic restriction; given aforementioned issues, however, we have little choice. Note that this doesn't prevent us from testing scaling (devices in sensor range), queries, etc.; what it means is that we'll have less *faces* on our map. Too bad, but it is what it is.



So first you can see the plane loading. Then the data loads, and after a brief loading delay, the map comes in from OpenStreetMaps. I'll zoom the camera in and out a bit; you can see that it's 3D, and the control interface works much like Starcraft or other real-time strategy games, except with people instead of alien troops. Now you can see I'll draw a box to select a group of data, and after a brief delay, the data and map will re-draw to allow more focus on the data in question. I can hover over various nodes to see their MAC addresses and locations, but for maximum data, I click on a node, and it shows me everything. I have some of the services I use, I have the hardware and software I'm carrying, I have a real name, email address, and even my photo from an online dating site. Combined with the true location and time of each of these pings, we end up with the same data that you used to use a whole team of surveillance agents to retrieve. Cheap, distributed stalking.

Roadmap

- Goals
- Background
- Architecture
- Design of CreepyDOL
- Future Work
- Mitigation

Scaling Up

- Sharding Contagion Networks
- Scaling backend --- luckily, this isn't hard
- Scaling limits of visualization

Sharding the contagion networks: it's easy, just give them different keys. Each network could have a sink node that throws data into the visualization system.
Scaling the backend is similarly easy: the software communications with the visualization engine over HTTP, so it can run in the ubiquitous cloud. Indeed, running the backend on Amazon S3, I've tested scaling parts of the backend to over half a terabyte of packet capture data.
The visualization is somewhat more difficult; Unity gets fussy if I display more than a couple thousand nodes at once. However, with grouping, and eventually, over large map areas, doing limited field of view and view distance work (as they do in real video games), this can be mitigated.

Enhancements

- $20 SDR devices (RTLSDR)
  - To listen to any frequency, not just WiFi
- Encrypted WiFi Workarounds
  - e.g., Reaver
- Jasager (WiFi Pineapple) to make sure wireless devices connect
  - MitM

Roadmap

- Goals
- Background
- Architecture
- Design of CreepyDOL
- Future Work
- Mitigation

Mitigation

- What do you want to sacrifice?
- Massive Leaks at all levels:
  - WiFi: Beaconing, constant pinging without being asked
  - OS: Seriously, we need enforceable VPNs in mobile OS (e.g., iOS)
  - App: Why do apps transmit so much data?
- This is everyone's fault. So we're kind of doomed.

So it's the status quo, right? Unfortunately, (next slide)

The Status is Not Quo

Image from Dr. Horrible's Sing-Along Blog, by Joss Whedon

We can't tolerate this level of privacy leakage: as consumers, we should demand better, and as developers at every level, we have a responsibility to do better.

Digression 2: Hark

- Archive for hacker work of all types (not just security)
- Mentorship, promotion, and archival forever
- New system of unique identifiers, like the academic DOI system, but free
- On Kickstarter now: http://thehark.net

Friday, August 2, 13

So a very short final note on Hark. There's been a back and forth between academic and non-academic researchers for years, where the academics say hackers aren't rigorous enough and don't cite their work, and hackers say academics don't do anything *but* cite other work. After this blew up at ShmooCon 2013, those of us who, like myself, straddle the academic/nonacademic divide, had some discussions and drew up plans for a way to let hackers archive their work, whether it's a tweet, a blog post, a conference presentation, or a journal article, and cite previous hacker work regardless of whether it's been academically published. I don't have time to go into all the details right now, but if you think it's important for hackers to stop re-inventing the same wheels every time we have a new research projects, I hope you'll check out thehark.net. And yes, we encourage corporate donations.

Thanks!

- To all those I've asked for comments, to Mudge for CFT, and my law school, for letting me spend so much time on other things.
- Also, I'm finishing law school in 10 months, and am wondering what I ought to take on next. If you've got something interesting, ping me: brendan@maliceafterthought.com.
- http://thehark.net