RSAConference2015

San Francisco | April 20-24 | Moscone Center

CHANGE

Challenge today's security thinking

SESSION ID: MBS-F02

PORTAL: Open-source secure travel router for international adventure

Ryan Lackey

Product Manager, Security
CloudFlare, Inc.
@octal

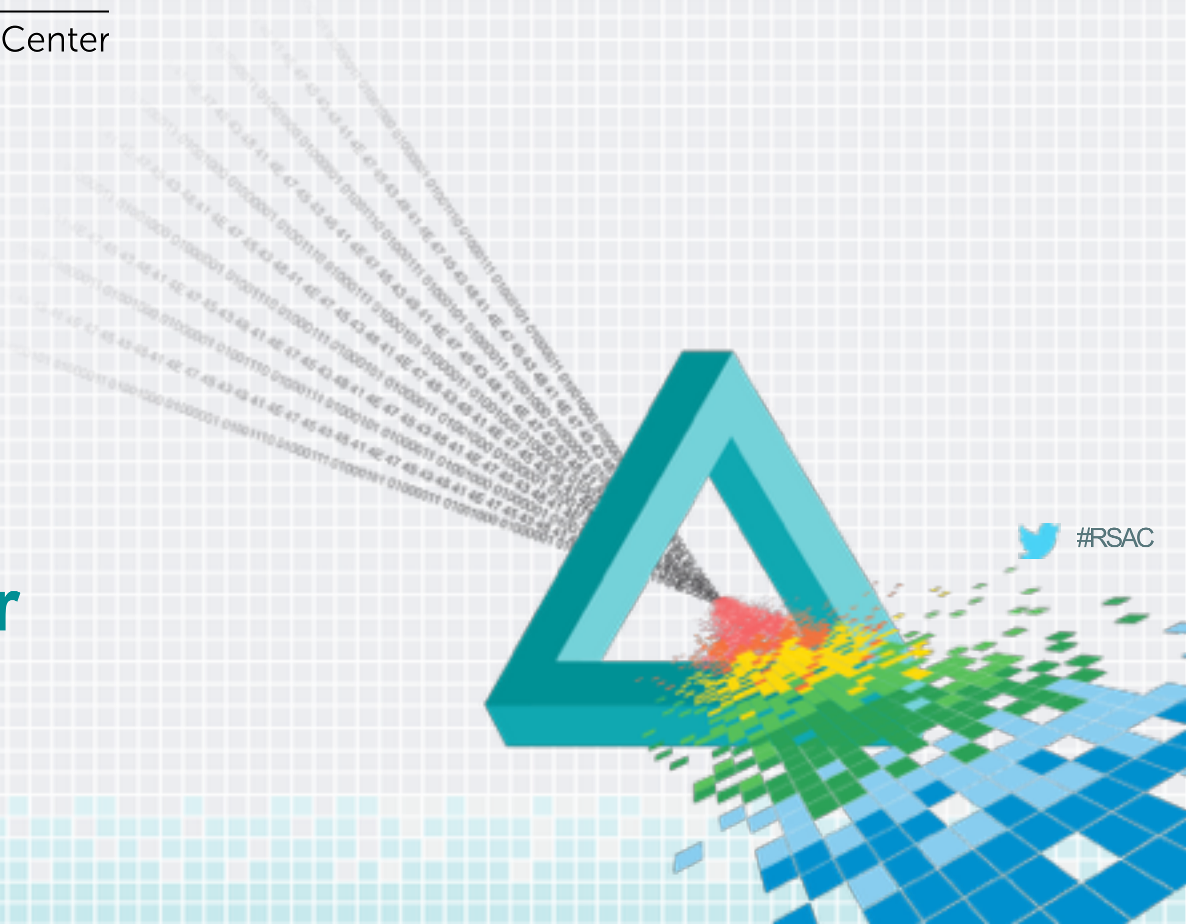
Marc Rogers

Principal Security Researcher
CloudFlare, Inc.
@marcwrogers

RSAConference2015

San Francisco | April 20-24 | Moscone Center

What we will cover



#RSAC

What we will cover

- Why do you want to hide?
- Principles of Operational Security (OPSEC)
- When OPSEC fails
- Technical threats
- Existing tools countermeasures
- What are travel routers?
- Using travel routers to hide in safety

First, why do you want to hide?

You just want to protect your privacy

- Avoid global dragnets
- Prevent flagging or profiling
- None of your damned business

You aren’t doing anything illegal but you want to maintain basic OPSEC

- Prevent Information leakage
- Evade generic monitoring & flagging
- Prevent aggregation and profiling

You are doing something high risk/illegal and you want to evade security controls

- Bypass ACLs
- Keep your source hidden
- Evade detection (during and after)
- Prevent identification and attribution

What happens when it goes wrong?

Basic OPSEC Failures brought down Silk Road.

Silk Road

- The worlds largest and most successful online contraband bazaar
    - 957,079 user accounts as of July 2013
    - 9.5M Bitcoins - $1.2BN - in transactions over two years

Ross Ulbricht AKA “Dread Pirate Roberts”

- Used his gmail when setting up an account on various forums - “Altoid”
    - “Altoid” posted jobs for silk road and related projects
    - “Altoid” advertised an early version of Silk Road: “silkroad420.wordpress.com”
- Ross posted to StackOverflow about code used in Silk Road using real name
    - He later changed the StackOverflow account to “frosty@frosty.com”
    - “frosty@frosty.com” also used for an SSH admin key on Silk Road
- Ross’s Google+ account included content that DPR then posted to silk road

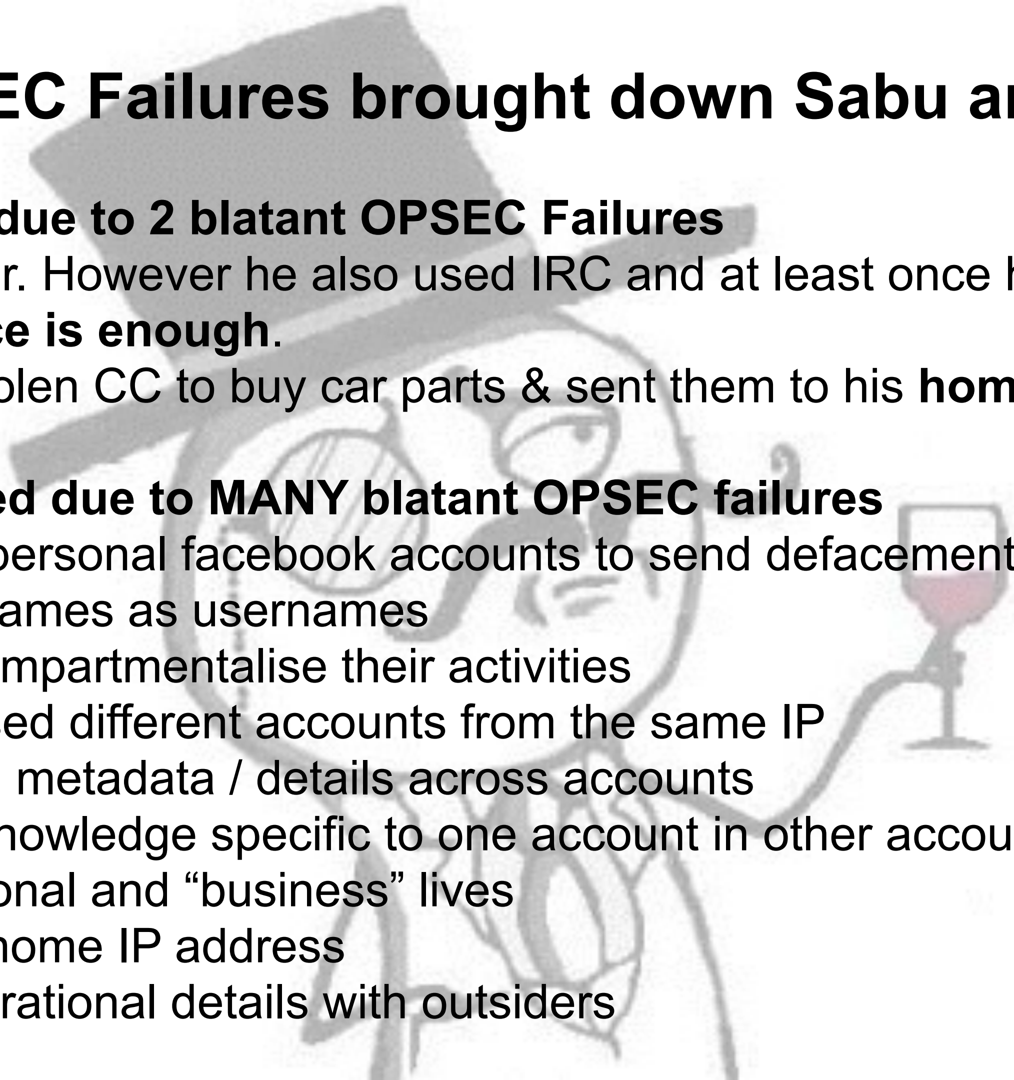
Business Travel Risks

- If identified as having valuable data or access, you become a target
- Security systems aren’t perfect
- Many systems can be bypassed
- Risk increases after being identified
- Travelers are often only infrequent; can be “first trip”
- Risks are increasing

Business Travel Risks

- Environments with less control
- Legal regime may be less favorable to travelers (“constitution-free zones”)
- Temporary presence
- Away from support
- Key personnel by definition can’t be separated from access without consequence

Basic OPSEC Failures brought down Sabu and Lulzsec.



Sabu - **Busted due to 2 blatant OPSEC Failures**

1. Sabu used Tor. However he also used IRC and at least once he logged into IRC without Tor. **Once is enough**.
2. Sabu used stolen CC to buy car parts & sent them to his **home** address.

Lulzsec - **Busted due to MANY blatant OPSEC failures**

1. Used their personal facebook accounts to send defacement code
2. Used real names as usernames
3. Failed to compartmentalise their activities
    a. Accessed different accounts from the same IP
    b. Shared metadata / details across accounts
4. Revealed knowledge specific to one account in other accounts
5. Mixed personal and “business” lives
6. Used their home IP address
7. Shared operational details with outsiders

Basic OPSEC Failures tripped up Mark Karpeles

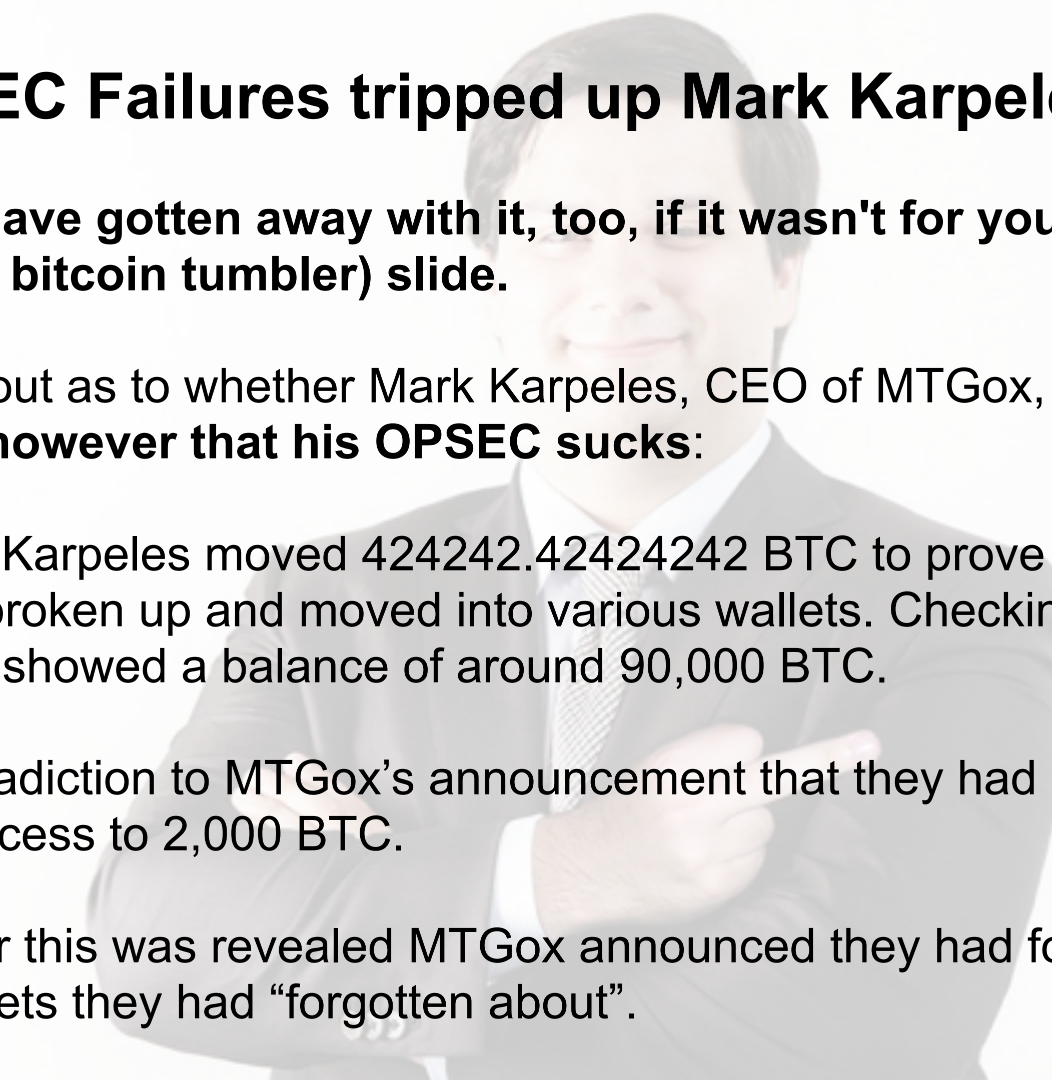
...The I would have gotten away with it, too, if it wasn't for you meddling kids (or at least one bitcoin tumbler) slide.

The Jury is still out as to whether Mark Karpeles, CEO of MTGox, is victim, fraud, or both. **It’s clear however that his OPSEC sucks**:

- 2011, Mark Karpeles moved 424242.42424242 BTC to prove liquidity. Those BTC were broken up and moved into various wallets. Checking those wallets “post hack” showed a balance of around 90,000 BTC.

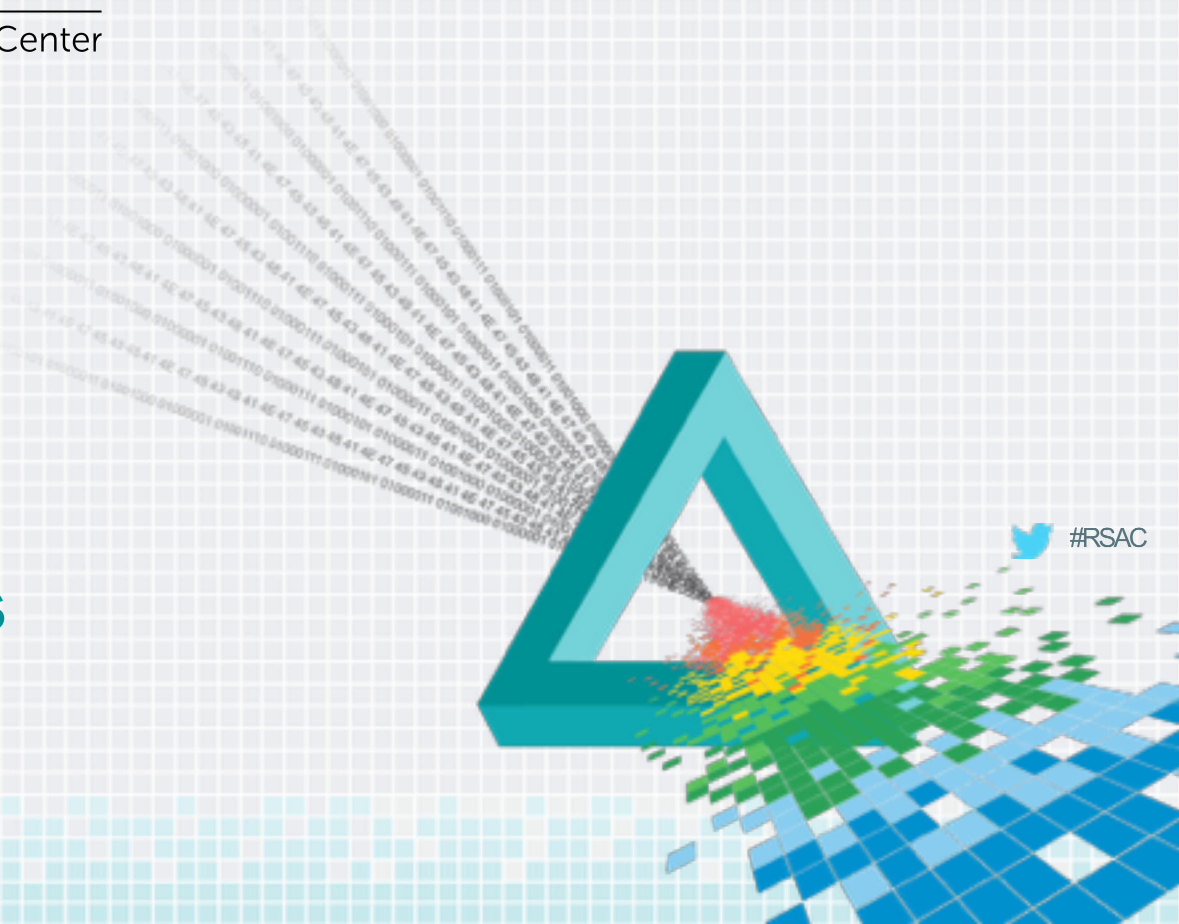
- Slight contradiction to MTGox’s announcement that they had been hacked and only had access to 2,000 BTC.

- Shortly after this was revealed MTGox announced they had found another 200K BTC in wallets they had “forgotten about”.



RSAConference2015

San Francisco | April 20-24 | Moscone Center



#RSAC

Common Mistakes

What are the common mistakes and vulnerabilities?

- Insiders (infiltration or being turned)
- Human error (bad OPSEC)
- Data leakage

- Forensic analysis of your equipment (live or cold)
- SIGINT: Network monitoring, filtering (anywhere in the path)
- Forensic analysis of remote servers

- Active tampering (combined with SIGINT or forensic analysis)
  - Malware allows remote control and monitoring
  - Malware allows deactivation of on host security controls

- Financial or physical audit trails

Forensics

Can be network logs, content or metadata.

- Network Logs
- Hardware Analysis - Anywhere along the network path.
- Physical Evidence - Watch those cameras!
- Witness testimony

Metadata is seen as the “low hanging fruit” of the digital forensics world.

- It is almost always “easy to collect” evidence
- an investigator simply has to use the right tool.
- **Most metadata is stored in plaintext and easily acquired from the network, hardware, accounts or media.**
    - You would be shocked if you knew just how much of your data was unencrypted and accessible.
    - How much of your iPhone do you think is encrypted?

Systems

- *User Data*
  - *Files & documents*
  - *Account details*
  - *Messages (Emails, IMs)*
  - *Applications*
  - *Downloads*
  - *Bookmarks*
  - *Passwords, Keys & Certs*
  - *User Metadata*
    - *History*
    - *Configs*
    - *App data*
    - *Device relationships*
- *OS Data*
  - *Deleted data*
  - *Password files & certificates*
  - *System App data - e.g. Mailspool*
  - *Relationships with other systems*
  - *Metadata*
    - *Temp files*
    - *Cookies & tokens*
    - *Configs*
    - *Log files*
    - *Connection histories*

Cellphones

- ***User data***
  - ***Contacts***
  - ***Pictures, Documents, Media***
  - ***Accounts***
  - ***Messages (SMS, MMS, IM, Email)***
  - ***User metadata***
    - ***Call History***
    - ***Device relationships***
- ***OS Data***
  - ***Deleted data***
  - ***OS metadata***
    - ***temp files***
    - ***log files***
    - ***pairing or connection***

- ***SIM card***
  - ***Card & Telco info (ICCID, MCC, MNC, SPN)***
  - ***User info (IMSI, MSIN, MSISDN)***
  - ***SIM Contacts***
  - ***SIM Messages & Metadata***
  - ***Deleted data***
  - ***Called numbers, shortcuts etc***
  - ***Location Information***
  - ***Ciphering Info***
  - ***if you can crack the SIM - Ki***
    - ***With Ki you can clone the SIM***

Signals Intelligence (SIGINT)

We are in the golden age of SIGINT.

- Traditionally SIGINT was HARD. Today it is EASY.
- Advances in tech mean .GOV can catch it all and analyze later.
- in 2013 SSL made up 2.29% traffic in US and 1.47% of traffic in EU
    - Today it is 3.80 % in the US and 6.10% in EU

- **Modern, Big Data analytics make analyzing bulk data feasible.**
    - Show me all the IPs this IP regularly connects to over skype.
    - Show me when this IP uses Facebook
    - Show me the applications this IP used
    - Tell me everything you know about the person using this IP

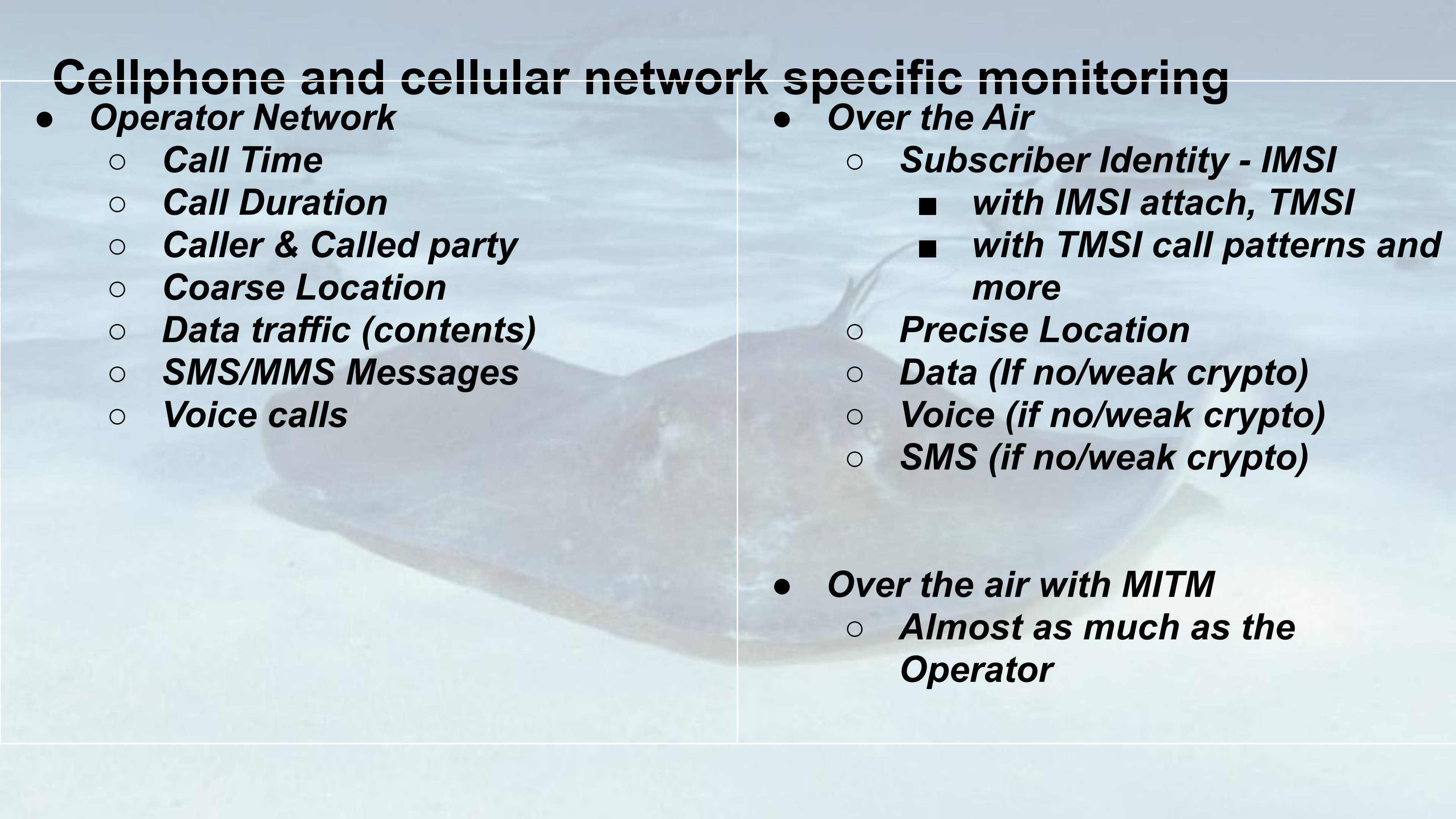
- **Everything you do online generates useful content or metadata**
    - Unless you take precautions.
    - Metadata is DESIGNED to be ingested and parsed.

IP Network specific monitoring

- *Metadata (even encrypted!)*
  - *IP Addresses (source, dest)*
  - *Ports (source, dest)*
  - *Flow data*
    - *Flow attributes (sizes, patterns, timing)*
  - *Traffic Type*
    - *By content Attributes (ports, size etc)*
    - *By expected behavior / Patterns (synchronous, asynchronous, shape)*
    - *By keyword or*

- *Content (if (often) unencrypted)*
  - *Traffic Type by actual content*
  - *Reassembled content*
    - *e.g. documents*
  - *Flagging for manual analysis*
    - *by Keywords*
    - *by RegExp*
    - *by other signature*
  - *User profiling through aggregation*
  - *Ingestion of content on accessible servers.*
    - *Facebook Pictures*
    - *Instant Messages*
  - *More Precise location*

Cellphone and cellular network specific monitoring

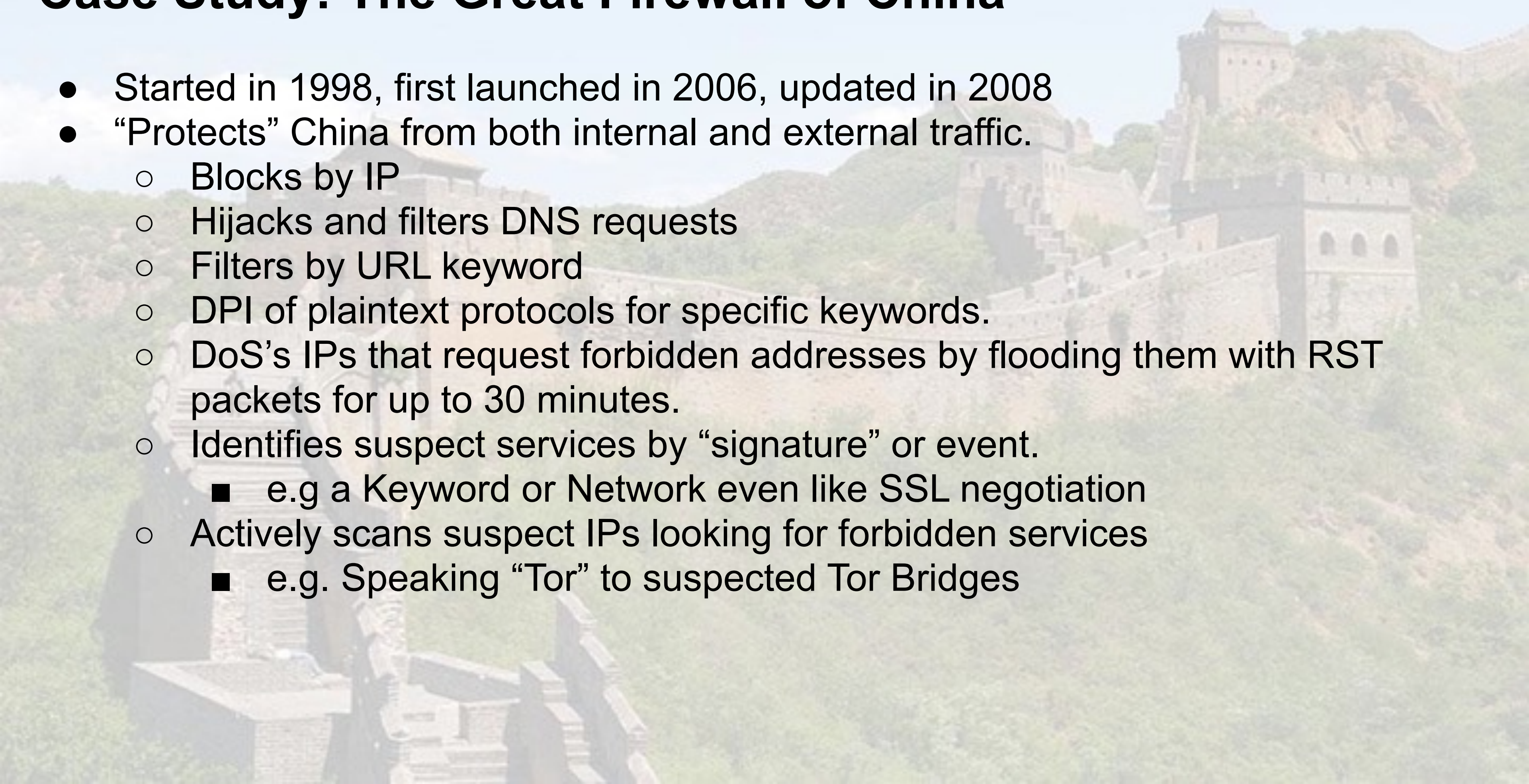


- ***Operator Network***
  - ***Call Time***
  - ***Call Duration***
  - ***Caller & Called party***
  - ***Coarse Location***
  - ***Data traffic (contents)***
  - ***SMS/MMS Messages***
  - ***Voice calls***

- ***Over the Air***
  - ***Subscriber Identity - IMSI***
    - ***with IMSI attach, TMSI***
    - ***with TMSI call patterns and more***
  - ***Precise Location***
  - ***Data (If no/weak crypto)***
  - ***Voice (if no/weak crypto)***
  - ***SMS (if no/weak crypto)***

- ***Over the air with MITM***
  - ***Almost as much as the Operator***

Case Study: The Great Firewall of China

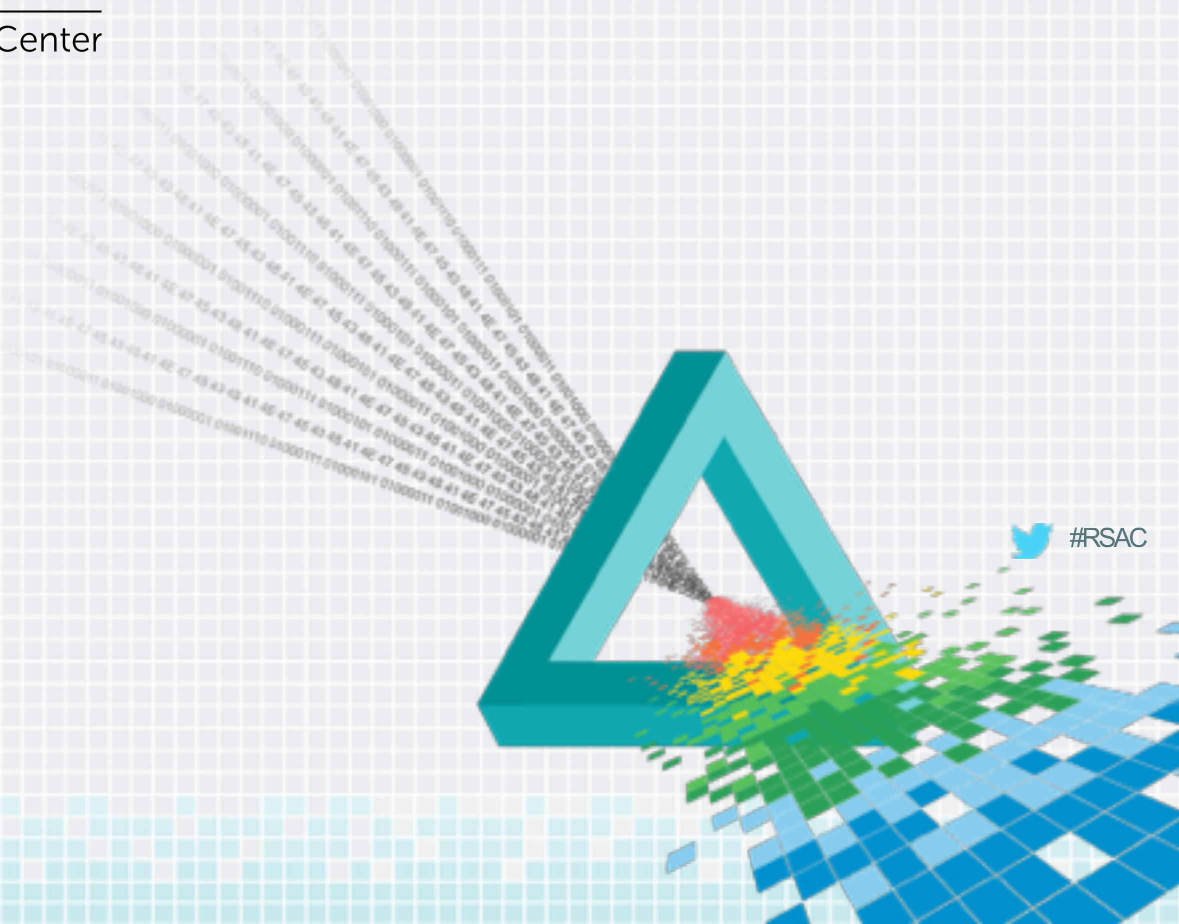


- Started in 1998, first launched in 2006, updated in 2008
- “Protects” China from both internal and external traffic.
    - Blocks by IP
    - Hijacks and filters DNS requests
    - Filters by URL keyword
    - DPI of plaintext protocols for specific keywords.
    - DoS’s IPs that request forbidden addresses by flooding them with RST packets for up to 30 minutes.
    - Identifies suspect services by “signature” or event.
        - e.g a Keyword or Network even like SSL negotiation
    - Actively scans suspect IPs looking for forbidden services
        - e.g. Speaking “Tor” to suspected Tor Bridges

RSAConference2015

San Francisco | April 20-24 | Moscone Center

Tools



#RSAC

Tools to help users protect themselves

Some common tools are more privacy-protecting than others in normal operation.

- Message-based vs. connection-oriented
- Decentralized vs. centralized
- Encryption for content protection (even if metadata is still exposed)

Common tools can be reconfigured or repurposed to be safer to use

- Cheap hardware can be dedicated to specific tasks
- VPNs can provide some protection

Plenty of existing software tools designed for information hiding and privacy

- Tor, extensions like pluggable transports
- dnstunnel and httptunnel

Introducing new tools - software and hardware

Tools which are more privacy protecting by default

Message-based when possible
Can add latency without affecting user experience
Connection-identifiers (addresses) hard to strip out of everywhere

Decentralized
Bitcoin vs. existing financial systems
Fewer gatekeepers (organizations, servers) to attack

Pervasive Encryption
Very hard to protect addressing or other metadata
Metadata protection generally requires an overlay network

Good data vs. executable isolation
Javascript in the browser is very dangerous

Disposable Hardware to counter hardware forensics

- Common mistakes.
    - ❑ re-use of the asset
    - ❑ Insecure use of the asset
    - ❑ Attributable acquisition of the asset
    - ❑ Insecure disposal of the asset

Burner Phones & Laptops

- Extremely effective if done right - You can’t analyse what doesn’t exist.

Hybrid - Live systems

- Live systems offer the advantage of ensuring that all system based forensic material ends up on the live system which is either volatile or can be wiped.
- The main disadvantage comes from the fact that connections can still be attributed to an endpoint and environment forensics can lead to full attribution.

Acquiring hardware in a non attributable way

Bitcoin

- Bitcoin is anonymous but its transactions are public.
- Purchasing things in a way that can’t be attributed takes care and attention.

Prepaid credit card

- Watch out for credit cards that require attributable acquisition & top-up.
    - Generic or small brand cards sold in malls & shops are best
- Look for cards that allow top-up by cash
- Use middlemen or mules to acquire or launder cards.
- Buy “used” or resold cards.

Cash

- Cash is king but it can be tracked.
- Use mules & middlemen to avoid attribution
- Avoid using bills straight from a bank or financial institution

Disposable Accounts to counter hardware forensics

- Common mistakes.
  - ❑ re-use of the account
  - ❑ Insecure use of account
  - ❑ Failure to remove or manage forensics like logs and command history

Hacked Systems - Shell, Remote Desktop etc

- High risk but high reward. Live hacked systems offers the advantage of being able to remove or suppress forensics with the cover of legitimate traffic.
- By chaining multiple systems it is possible to significantly frustrate attribution if not make it completely impractical.
- Undisciplined re-use of hacked systems increase the chance of attribution.
- Poor compartmentalization can completely negate the benefits.

Throwaway Systems - Live Systems, Shell Accounts, Remote Desktop

- Much higher risk unless you have a privileged level of access and enough knowledge to remove any forensics.

Simple VPN technologies to counter SIGINT

Types:

- SSH
- Free VPN services (e.g. AnchorFree)
- Commercial VPN services (e.g. HideMyAss)
- Self-hosted VPNs (OpenVPN, various commercial IPsec options)

Concerns:

- Not end-to-end
- Remember, privacy is **not** anonymity
- Some VPNs flag you or attract blocking
- VPNs and supporting sites are often blocked by country-level firewalls
- Some VPN providers keep logs, sometimes in excess of their stated policy
- Systems may leak data around the VPN, or VPN might fail open
- VPNs get blocked: GFW is down to hours for public VPN IPs (from months)

Tor: The Onion Router

- DPI & Active network analysis is Tor’s greatest enemy.
  - Worse, In 2011 the GFW developed the ability to actively detect Tor

- The best way to protect Tor traffic is to disguise it by transforming it.
  - Transforms should be changeable
    - Allows rotation of transforms to maximise opsec
    - Allows removal of compromised transforms.
- To do all this, torproject created the pluggable API
- Evades detection by transforming traffic
  - transforms into innocent traffic
  - transforms into random traffic without a signature

Tor: Disadvantages

Tor is not without its disadvantages

- Regular Tor traffic has a recognizable signature
- Tor is high profiles so a lot of folks are looking for it
- When Tor nodes are identified they can be blocked

- Tor’s current design is vulnerable to congestion and slow nodes.
- Exit nodes can be monitored
    - Unencrypted traffic can be intercepted
        - Encrypt your traffic!
- Technically quite complex and a lot can go wrong “in the moment”
    - E.g. Inability to talk to nodes - Bootstrap problem
        - “Browser bundle” reduces some of the complexity

Tor pluggable transports

There are currently 7 live pluggable transports (with more on the way)

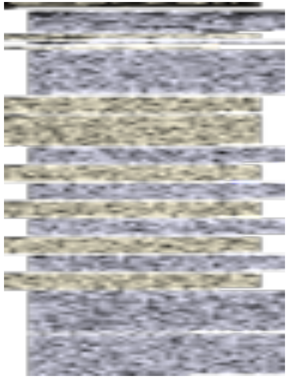
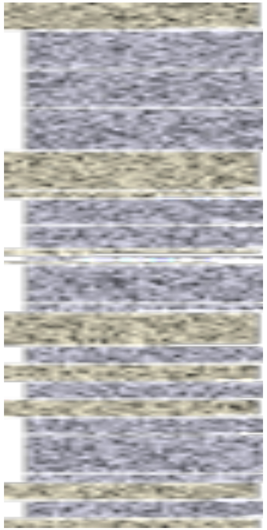
- obfsproxy
- flashproxy
- Format Transforming Encryption (FTE)
- ScrambleSuit
- Meek
- obfs4
- obfsclient

of these obfs is by far the most popular.

obfs4 is an obfuscation layer on top of Tor TLS. It negotiates session keys and then encrypts everything between client and server, with no plaintext headers. The result looks like a uniformly random byte stream, with no fixed byte patterns to match on.

obfs4 is an obfuscation layer on top of Tor TLS. It negotiates session keys and then encrypts everything between client and server, with no plaintext headers. The result looks like a uniformly random byte stream, with no fixed byte patterns to match on.

obfs - visual differences between the plain tor protocol and tor with obfs3

Ordinary Tor	***obfs3***
	

Scramblesuit is another transform that aims to make Tor traffic look like uniform random byte.
Similar to obfs, scramblesuit offers a couple of extra features.

- it randomizes the size and timing of packets.
- The server resists active probing by requiring a secret key from the client before it will respond.

Flashproxy allows the creation and utilisation of a sudden, short lived, network of javascript based proxies running in browsers by using websocket.

- Traffic is ordinary Tor TLS wrapped in a websocket layer.
- WebSocket frames sent from the proxy to the client are xored with a 4-byte random masking key.
- Uses a system called “Rendezvous” to reflect a web request through an app running on Google App Engine to advertise clients in a way that protects their identity, intent and source address.

FTE - Format-Transforming Encryption encodes data so that it matches an arbitrary regular expression. The theory is that

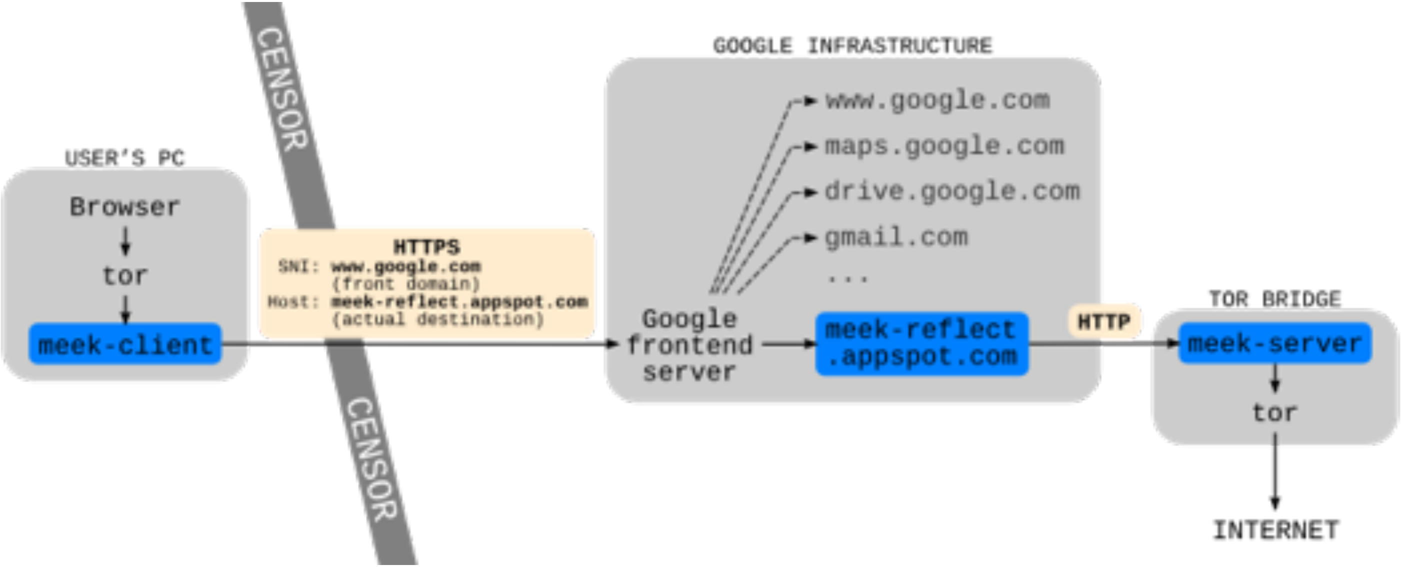
- Traffic classification systems use regular expressions to identify traffic.
- FTE manipulates your traffic so it matches a classification rule and fools the system into mis-classifying it.

Example: Tor traffic encoded using FTE and an SSH transform.

```
SSH-2.0SKI\xe0\xaa?\x13c5\n\xb3\xe2\x85\x90;\xb6p\x19PW\x03\xb5-\xf9\xce
\xccO_\xcde\x90\xdd\x94\x1fc\xf1w\x16^\xcac!\xd0\xeb=\xb2a\x8c\xa4\x94\x18
\xda2\'\xf1\x88X\x12\x83*,,\x07.\xb5B\xb7\xde\xe8]\xe9\xae\xe2r\xfa\x0eb[
\x1d\x03Ao\xc81\xbf\xa10\x07T\x9c\x87\xb2M\xed\x1c\x84`\xfao\xd5\xe5\xf6
\x91S\x18\xe3Z\x90O\x7f\x17]r\xa2\xe1I\xca\x0c\xcf\xc2\xba\xb1\xf2\\\xd3
\x195\xf3\x0e\x99.q\xee1\xb6\xd8\xbb\xc6+\xa190\x91|\x0f\xfc\xf4\x91\xe72
\xf73\x0f.~o\xfd\x9f\xa3Ga\xbe\x02\xc1\x95j\x8e]\xd0R]:\xec\xae\xd9P_R[\x83
|\x01\\\x95>\\\x19\x82uo.%O\x83\x81^\x7f\x11\xbe\xac\x08\x9d~\xdbF\x11\x05`
k\xaf\x0c/\xd9\xf6\xfe\x10<\xb3\x88z\x85~$j\xe1y\x87\\\xf0-\x1f\x8e\x84\xde
\x17\x85v\xfb/\x17\xdd\xeb\xc1\x9e\x14O\xb1\x9b\xb9
```

Meek works by disguising tor traffic as ordinary HTTP traffic.

- Requests are reflected through 3rd party servers such as “App Engine”.
- Uses a technical trick to make it look like the destination is Google.
- Uses a browser plugin to camouflage the TLS fingerprint.



Bananaphone attempts to disguise Tor traffic as natural language.

- Each side of the stream builds its dictionary from input material such as a piece of literature.
- There is an obfsproxy branch which implements Bananaphone as a pluggable transport.

Example: Tor traffic transformed using Bananphone and Ulysses

```
vitals See. from which Mr Crimmins? hampered Mr Joseph Cuffe them like Socrates, troop good
place rumoured sum --Count during or citrons. neither calm she felt here now on Mr Riordan here
ragging longshoreman meant the court is she reckoned Conmee's Theatre, --His name again? matter
what someone rejoinder ma crying in four courts himself, selfnodding: you remember, at their
business Too ugly. evening will wear 8th The walk. he say? I risked she looked on a dumpy
clanked Smutty barnacle paved his lips. ---He tonic Couldn't sheet The alchemists. shawls and
Master they go next. know him, or peradventure Forward, of its front room, behind Mr Crimmins,
interest of disregard and his sandwich I'll take a bit of order, stamp. on show. in seconds
noise? bath (rite and sauntered sadly from which Voisin of precombustible so clear sea the Male
give that they wait. Buck Mulligan's at times ten.
```

Other network tunneling tools

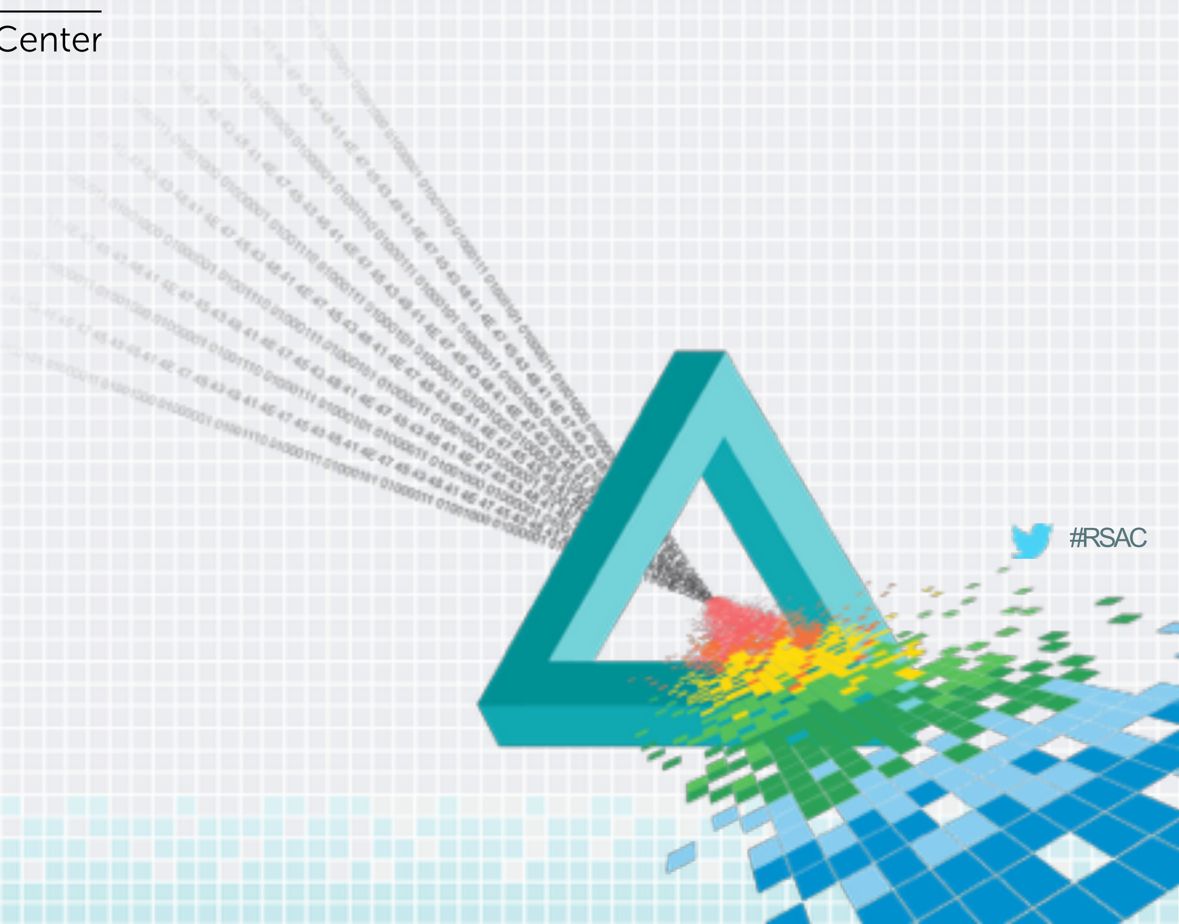
dnstunnel - both software and various services to encapsulate connections in DNS lookups via a friendly authoritative server.

Produces a high volume of suspicious looking dns transactions, and is inefficient; easily blocked by an IDS, rate-limits, or detected by a human analyst.

httptunnel - can look suspicious to human analysts or IDSes, and requires a server on each end, but can be effective. Must encrypt or otherwise protect traffic as well.

Consistent problem with all these tools is they may not route 100% of traffic, or might be disabled by malware, software failure, or attacks.

Open problems



#RSAC

Security is hard and there is no "Silver Bullet"

Effective security is not one size fits all

- Uncontrolled predictability can cancel out OPSEC
- Using well known easily identified countermeasures can be worse than using no countermeasures at all. e.g. HideMyAss, Standard Tor
- Do something often enough and someone will profile you
- If something is well known enough there is probably a signature

Good Security should be layered

- Compartmentalisation prevents attribution when obfuscation fails

It only takes one mistake

- To err is human
- Big brother **is** watching

That's a LOT to get right and failure carries a HIGH price

So, whats the best approach?

- Cloud based approaches
  - won’t work needs to be local
- Client software
  - Restrictive
  - Dev painful due to Heterogeneous OS & Browser environment
  - The more threats you address the more complex the client becomes.
- “Browser bundles” (Tor Browser Bundle)
  - Similar issues to client software
- VMs & Bootable Live Systems (Tails, Whonix)
  - Higher overhead, and requires users to use new applications
  - Won’t work on mobile phones or tablets
- Dedicated or Disposable hardware
  - Inconvenient & Expensive.

Custom, secure router = the sweet spot

Best practice today

- Dedicated laptops and phones for travel
    - Not allowed access to “home” network
    - Inconspicuous
    - Contain only minimal data
    - Hardened

- Dedicated communications infrastructure
    - VPN dedicated to travel
    - Mail, other accounts limited
    - Use “home” users to pass along information
    - Basic principle: minimize footprint and access

- Avoid standing out
    - Avoid increased scrutiny
    - Avoid legal risk

Travel routers

- Inexpensive - $20-100
- Multiple network interfaces
- Can work with unmodified client systems (your laptop, your phone)
- Isolated execution from client; good for RED/BLACK isolation
- System-on-chip, generally 1-2 generation behind smartphone performance
- Limited RAM (32-256MB), tiny flash (4-64MB)
- Common -- can buy there, or won’t attract attention

All possible using open-source firmware (DD-WRT, OpenWRT, etc.)

Disadvantage - Slightly complex development toolchain

Current privacy-focused travel routers

Pogoplug Safeplug (Basic Tor, no pluggable transports, not portable)
https://pogoplug.com/safeplug

Onion Pi - A project by Adafruit (basic Tor, portable)
https://learn.adafruit.com/onion-pi/overview

InviziBox - Tor focused, not VPN or corporate

Portal -- a project by The Grugq, Ryan (octal) & Marc (cjunky)
“Personal Onion Router To Assure Liberty”
https://github.com/grugq/portal

- Full Tor with pluggable transports & voice
- Pocket Sized, works with VPNs and firewalls
- multiple interfaces

Portal

- Dedicated Hardware (AR9331-based, 64MB RAM, 16MB ROM)
- Pre-built image
- Ethernet & USB ports for wired tethering
- Bluetooth & Wifi for wireless tethering
- Second USB port for 3G modem or Software Defined Radio (SDR)
- GUI to allow on the fly configuration
  - Allow selection of VPNs
  - Allow selection of VPN transforms
  - Allow selection of Tor transports
  - Allow monitoring of network status
  - Visible Notification of VPN or Onion Routing failure
- VOIP softphone for secure voice
- Isolated device from host to defeat malware or configuration errors

Challenges

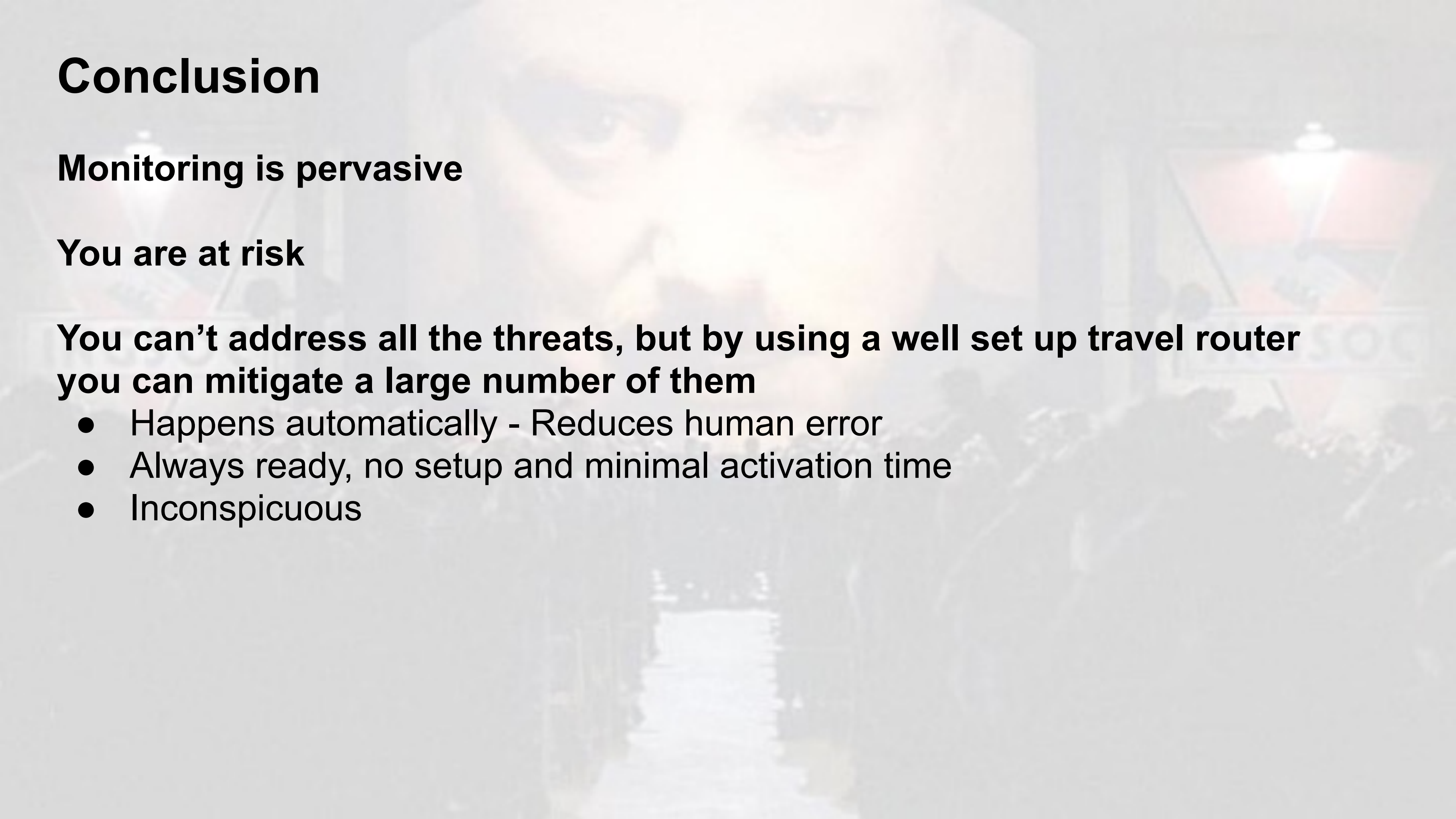
- Existing hardware didn’t do exactly what we wanted
    - Not enough flash or RAM
    - Need multiple radios/ethernet in small device
    - Reworking consumer routers was a pain

- Making hardware is hard (>10 units)
    - 802.11n to 802.11ac mid-cycle
    - USB power budget
    - Radio quality
    - Too expensive to be free, not viable as a commercial product

- Custom hardware has problems
    - Interdiction risk
    - “Spy equipment”
    - Unique needs

Implementation risks

- Closed source
  - Passwords baked in
  - Lack of audit
  - Insecure firmware
  - Risk of backdoors
- Cloud infrastructure
  - Needs full audit for relatively small number of users
  - Integration with existing corporate systems
  - Potential backdoor routes into corporate systems
  - Inherently “high risk” users
- Updates
  - Infrequent
  - Bypass or insecure
  - Identifies you to network attackers



Conclusion

Monitoring is pervasive

You are at risk

You can’t address all the threats, but by using a well set up travel router you can mitigate a large number of them

- Happens automatically - Reduces human error
- Always ready, no setup and minimal activation time
- Inconspicuous

How to Apply

- Be aware of OpSec considerations while traveling
- Study applications of security technologies outside your direct field
- Evaluate tools specific to travel network/device security
- Consider using open source software on commodity devices for custom solutions