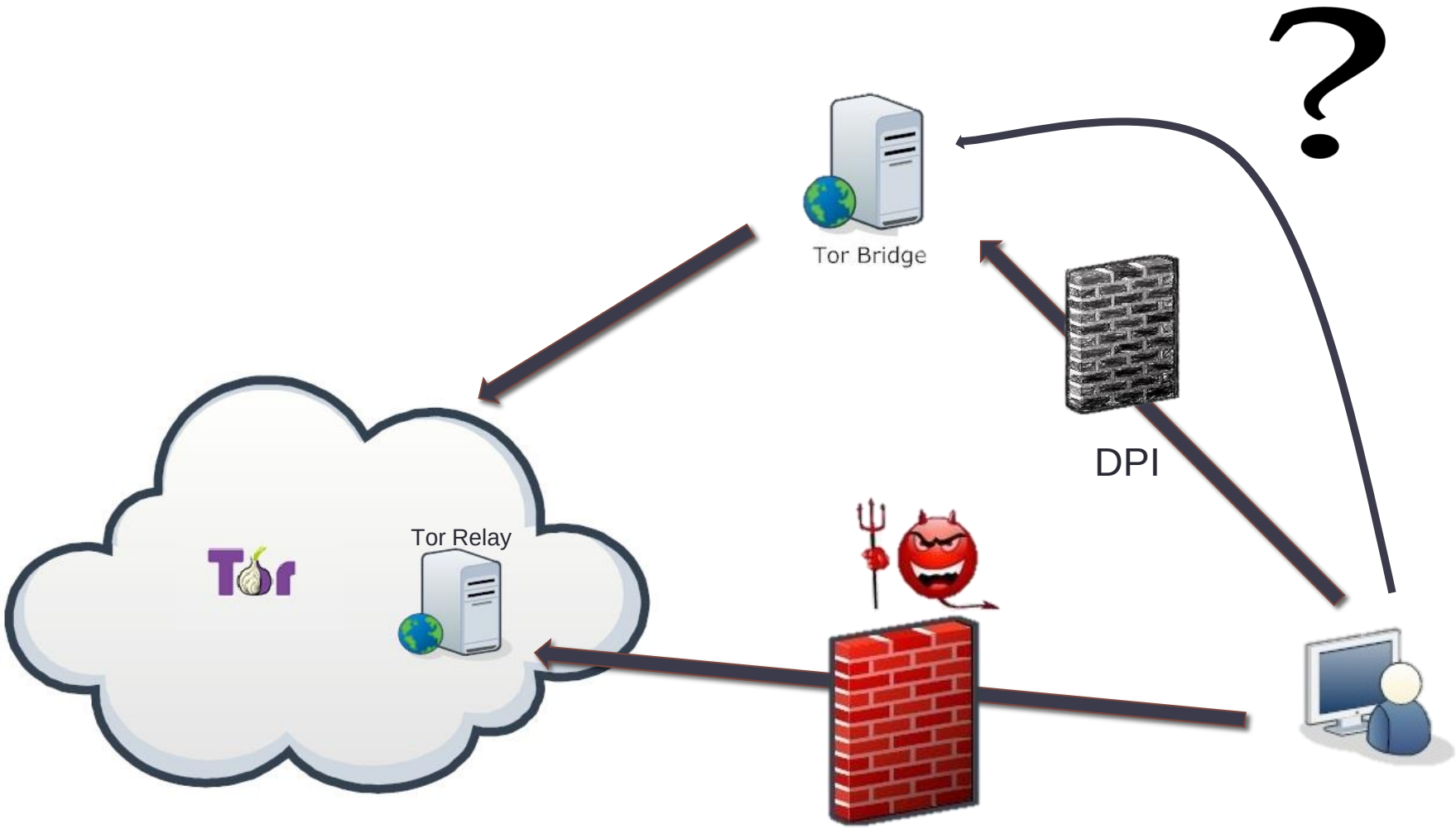
THE PARROT IS DEAD: OBSERVING UNOBSERVABLE NETWORK COMMUNICATIONS

Amir Houmansadr, Chad Brubaker, Vitaly Shmatikov
The University of Texas at Austin

IEEE Symposium on Security and Privacy (Oakland) 2013

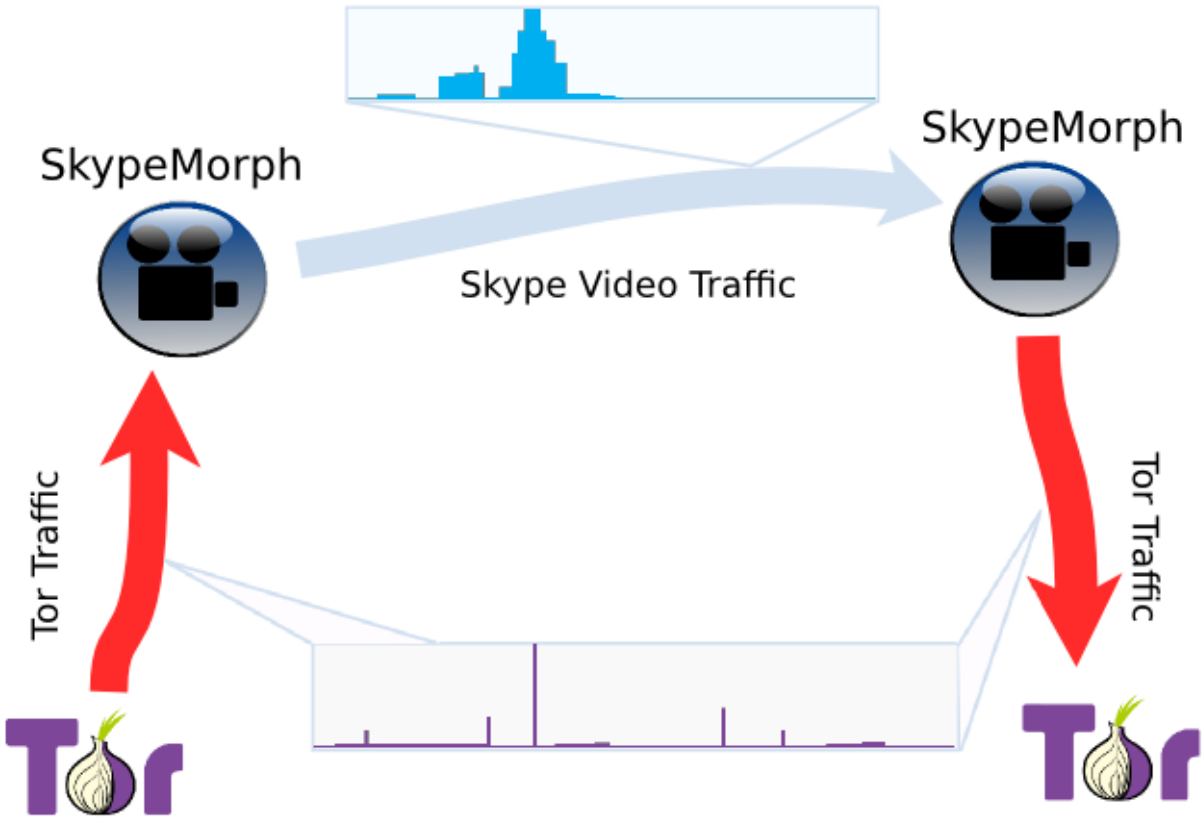
Motivation



Background and Contributions

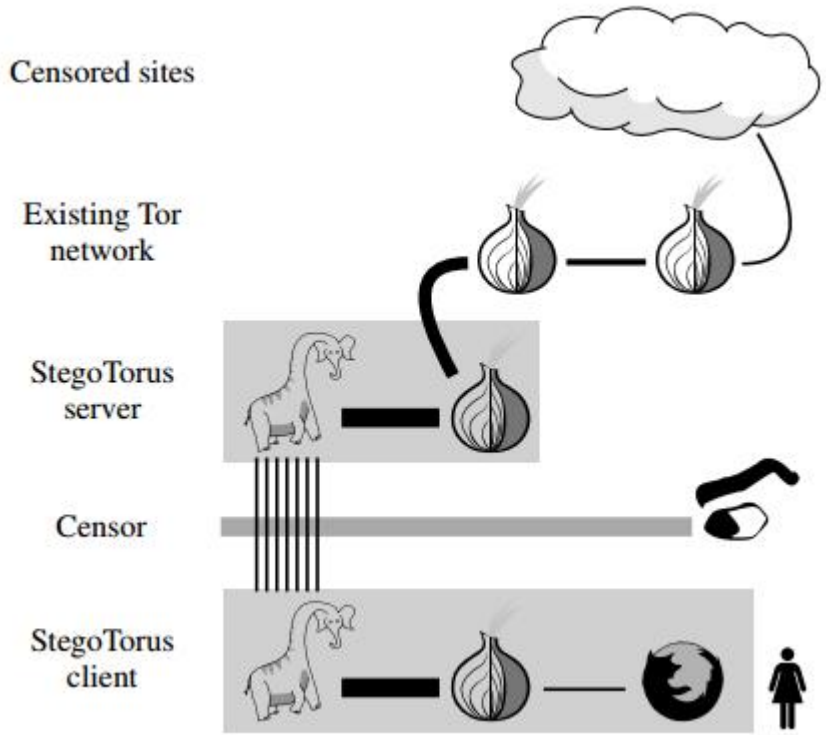
- Parrot circumvention systems
  - Uncensored target protocol
  - Unwillingness of censor due to collateral damage (economic reasons, etc.)
- Main Contribution:
  - Taxonomy of parrot adversaries
  - List of technical requirements that a parrot system must satisfy to successfully mimic another protocol
- Premise:
  - ***“Unobservability by imitation” is fundamentally flawed.***

SkypeMorph



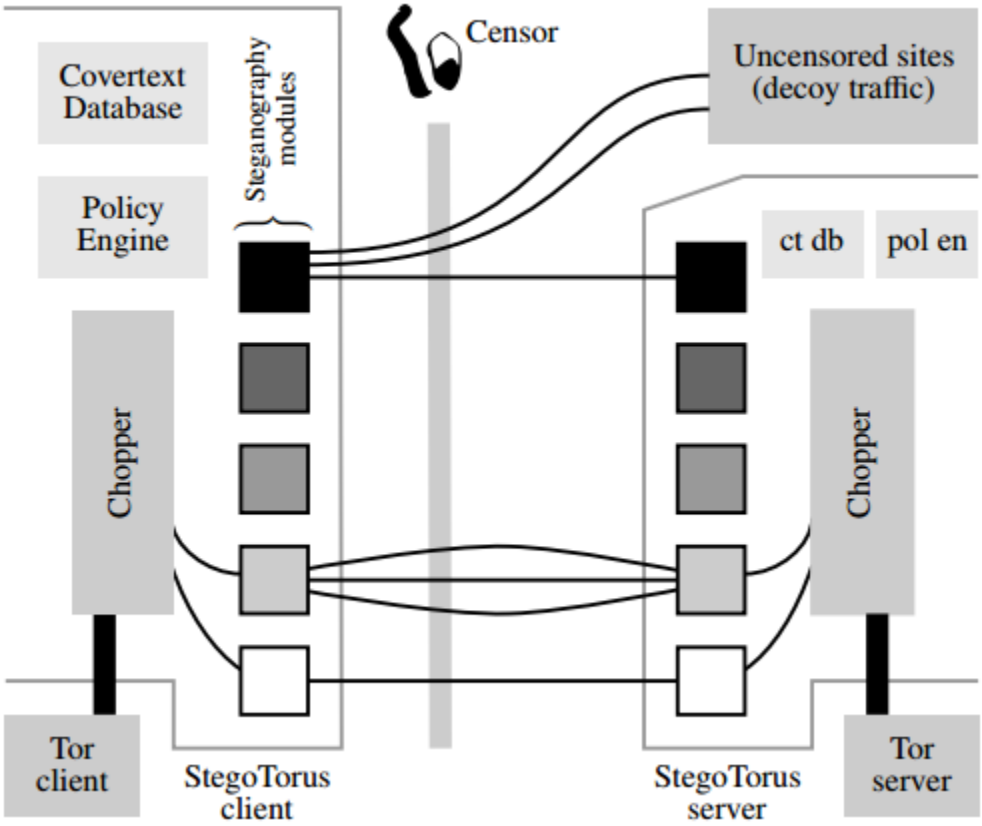
H Mohajeri Moghaddam et al, SkypeMorph: Protocol Obfuscation for Tor Bridges. In CCS, 2012.

StegoTorus



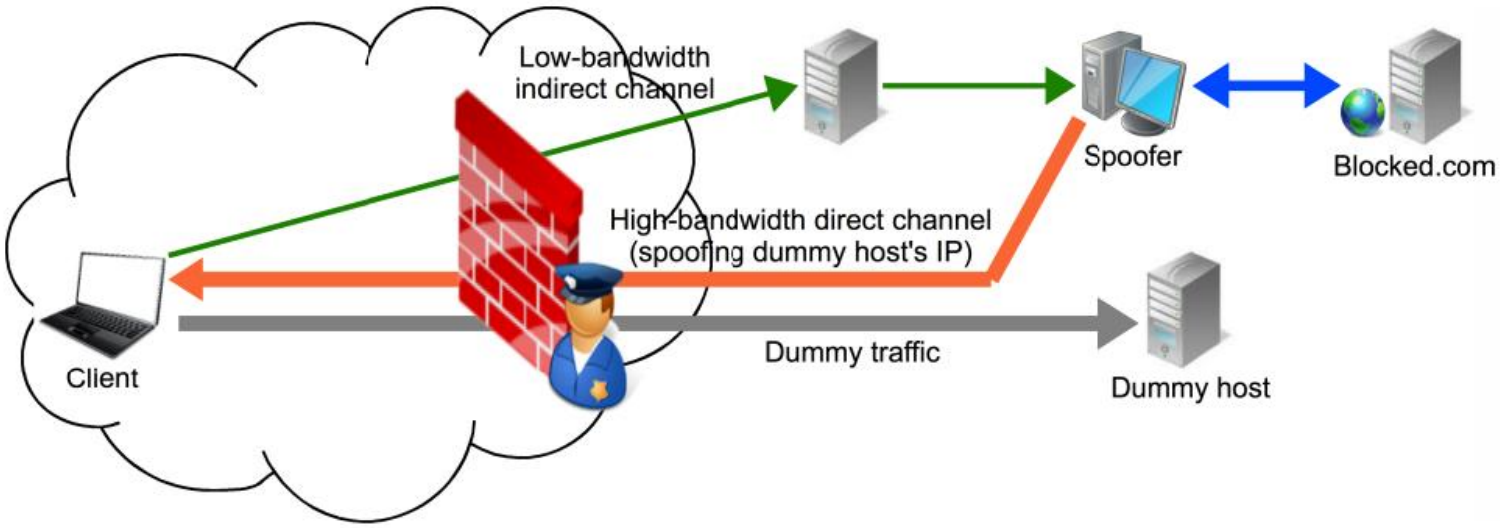
Z. Weinberg et al. StegoTorus : A Camouflage Proxy for the Tor Anonymity System. In CCS, 2012.

StegoTorus (cont'd)



Z. Weinberg et al. StegoTorus : A Camouflage Proxy for the Tor Anonymity System. In CCS, 2012.

CensorSpoofer



Q. Wang, et al. CensorSpoofer: Asymmetric Communication Using IP Spoofing for Censorship-Resistant Web Browsing. In CCS, 2012

Adversary Models

- Capability Classification
  - Passive
  - Active
  - Proactive

- Knowledge classification
  - Local (LO)
  - State-level oblivious (OB) : StegoTorus
  - State-level omniscient (OM) : SkypeMorph, CensorSpoofer

Parrot Requirements

- Mimic Protocol Entirely:
  - Correctness
  - Side Protocols
  - Intra Dependencies
  - Inter Dependencies
- Errors and Network Conditions:
  - Errors
  - Network Conditions

Parrot Requirements Cont'd

- Typical traffic
  - Content
  - Patterns
  - Users
  - Geographical characteristics
- Implementation Specific Artifacts:
  - Soft
  - OS

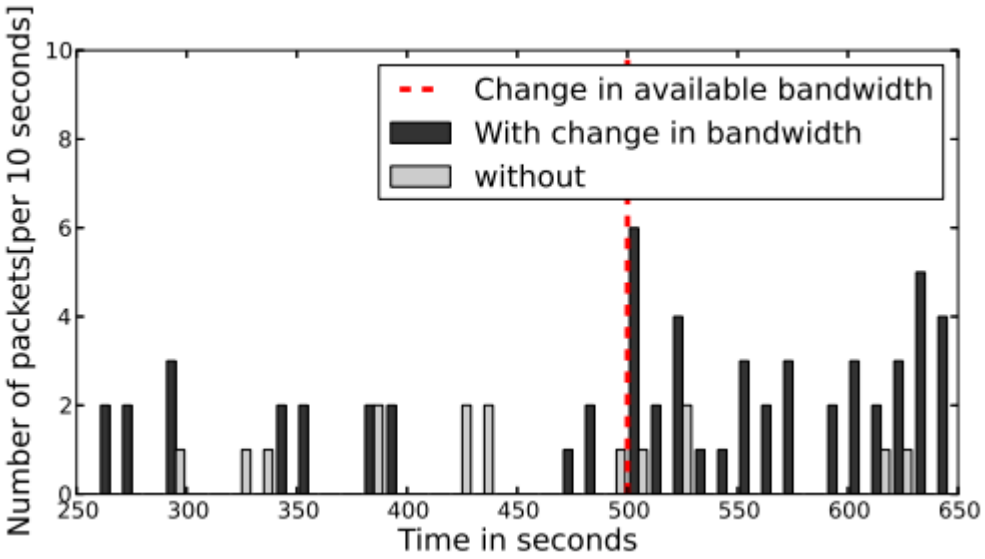
Attacks on Skype Parrot

- Passive Attacks:
  - Deviations from genuine Skype behavior
  - Re-use of pre-recorded Skype traces
  - Re-use of client-generated Skype traces

Attack	Imitation requirement	Adversary	SkypeMorph	StegoTorus-Embed
Skype HTTP update traffic (T1)	`SideProtocols`	LO/OB/OM	Satisfied	Failed
Skype login traffic (T2)	`SideProtocols`	LO/OB/OM	Satisfied	Failed
SoM field of Skype UDP packets (T3)	`Content`	LO/OB/OM	Failed	Failed
Traffic statistics (T4, T5)	`Pattern`	LO/OM	Satisfied	Satisfied
Periodic message exchanges (T6, T7)	`SideProtocols`	LO/OB/OM	Failed	Failed
Typical Skype client behavior (T8)	`IntraDepend`	LO/OM	Failed	Failed
TCP control channel (T9)	`SideProtocols`	LO/OB/OM	Failed	Failed

Attacks on Skype Parrot Cont'd

- Active and Proactive Attacks on Improved SkypeMorph and StegoTorus:
  - Verifying supernode behavior
  - Manipulating Skype calls
  - Manipulating the TCP control channel

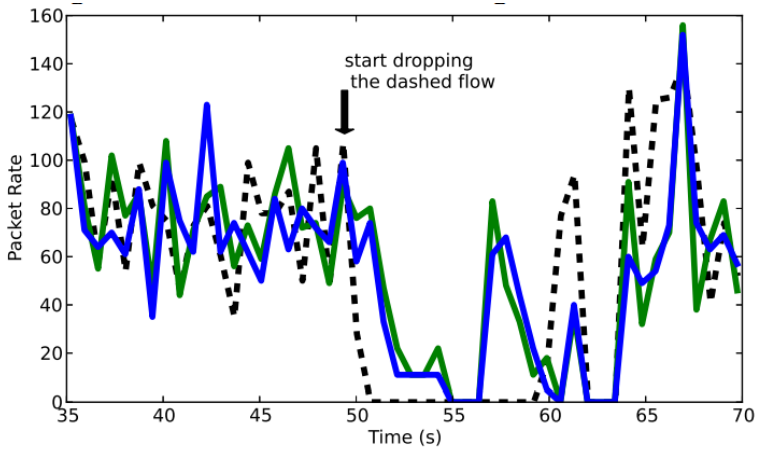


Attacks on Skype Parrot Summery

Attack	Imitation requirement	Adversary	Skype	SkypeMorph+ and StegoTorus+
Verify supernode behavior by flushing supernode cache	`SideProtocols` `IntraDepend`	Proactive, LO/OM	The target node serves as the adversary's SN, e.g., relays his Skype calls	Rejects all Skype messages
Drop a few UDP packets	`Network`, `Err`	Active, LO/OB/OM	A burst of TCP packets on the control channel (Fig. 1)	No reaction
Close TCP channel	`IntraDepend`, `SideProtocols`	Active, LO/OB/OM	Ends the UDP stream immediately	No reaction
Delay TCP packets	`IntraDepend`, `SideProtocols`, `Network`	Active, LO/OM	Reacts depending on the type of TCP messages	No reaction
Close TCP connection to a SN	`IntraDepend`, `SideProtocols`	Active, LO/OB/OM	Client initiates UDP probes to find other SNs	No reaction
Block the default TCP port for TCP channel	`IntraDepend` `SideProtocols`	Active, LO/OB/OM	Connects to TCP ports 80 or 443 instead	No reaction

Attacks on Skype StegoTorus

- Chopper Attacks
  - Correlating IP addresses between links.
  - Exploiting connection dependencies
- Passive attacks on StegoTorus-HTTP
  - Exploiting discrepancies in file-format semantics.
- Active and proactive attacks on StegoTorus-HTTP
  - Fingerprinting HTTP server
  - Manipulating HTTP requests



Attacks on Skype StegoTorus Summery

HTTP request	Real HTTP server	StegoTorus's HTTP module
GET existing	Returns "200 OK" and sets `Connection` to `keep-alive`	Arbitrarily sets `Connection` to either `keep-alive` or `Close`
GET long request	Returns "404 Not Found" since URI does not exist	No response
GET non-existing	Returns "404 Not Found"	Returns "200 OK"
GET wrong protocol	Most servers produce an error message, e.g., "400 Bad Request"	Returns "200 OK"
HEAD existing	Returns the common HTTP headers	No response
OPTIONS common	Returns the supported methods in the Allow line	No response
DELETE existing	Most servers have this method not activated and produce an error message	No response
TEST method	Returns an error message, e.g., "405 Method Not Allowed" and sets `Connection=Close`	No response
Attack request	Returns an error message, e.g., "404 Not Found"	No response

Attacks on CensorSpoofer

- Manipulating the tag field
- SIP probing:
  - Send a SIP INVITE
  - Send an invalid SIP message
  - Send a message for a non-existing call
- Manipulating upstream packets

Attack	Imitation requirement	Adversary	Typical SIP clients (e.g., Ekiga)	CensorSpoofer
Manipulate `tag` in SIP OK	`Soft`	LO/OB/OM	Nothing	Client closes the call
SIP INVITE to `fakeID@suspiciousIP`	`SideProtocols` `Soft, Err`	LO/OB/OM	Respond with "100 Trying" and "180 Ringing", "483 Busy Here", "603 Decline", or "404 Not Found"	Nothing
SIP INVALID	`SideProtocols,Err`	LO/OB/OM	Respond "400 BadRequest"	Nothing
SIP BYE with invalid `SIP-ID`	`SideProtocols` `Soft, Err`	LO/OB/OM	Respond "481 Call Leg/Transaction Does Not Exist"	Nothing
Drop RTP packets (only for confirmation)	`SideProtocols` `Soft, Network`	LO/OB/OM	Terminate the call after a time period depending on the client, may change codec in more advanced clients.	Nothing

Lessons and Recommendations

- Understand the capabilities of the adversaries
- Unobservability by imitation is a fundamentally flawed approach
- Partial imitation is worse than no imitation at all.
- One promising alternative is to not mimic, but run the actual protocol

Weaknesses…

- Specific flavor of protocols are targeted
- No solid methodology/algorithm for detection
  - TCP changes due to UDP fluctuation
  - Exploiting connection dependencies in StegoTorus
- Vague Methods:
  - Manipulating tag field in CensorSpoofer, or Supernode identification
- Arms race nature of Censorship Resistance

Questions?