Infranet: Circumventing Web Censorship and Surveillance

Nick Feamster, Magdalena Balazinska, Greg Harfst, Hari Balakrishnan, David Karger
M.I.T. Laboratory for Computer Science

Outline

- Motivation
- Other Approaches
- System Architecture
- Communication Protocol
  - Hiding
  - Tunnel Protocol
  - Modulation
- Security Analysis
- Performance Evaluation

Motivation

"We have a really serious problem in terms of the American free speech idea, but it is very American to make money. Between anticensorship and the desire to make money, the desire to make money will win out."

-- Jack Balkin, New York Times 11/19/2001

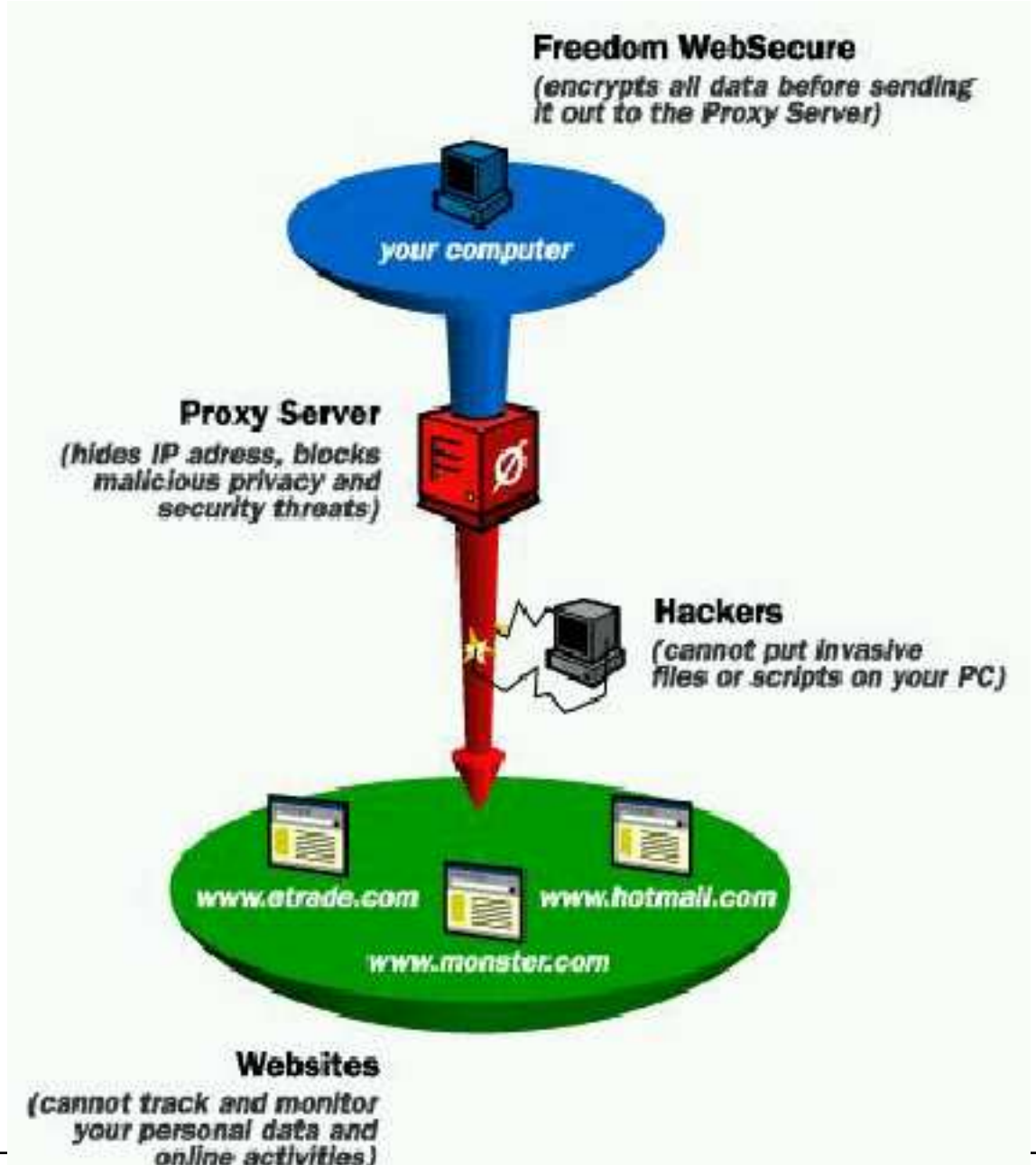
"[China] also tries to maintain strict control over what its citizens read on the net, blocking access to certain internet domains such as bbc.co.uk"

-- BBC News, 2/11/2002

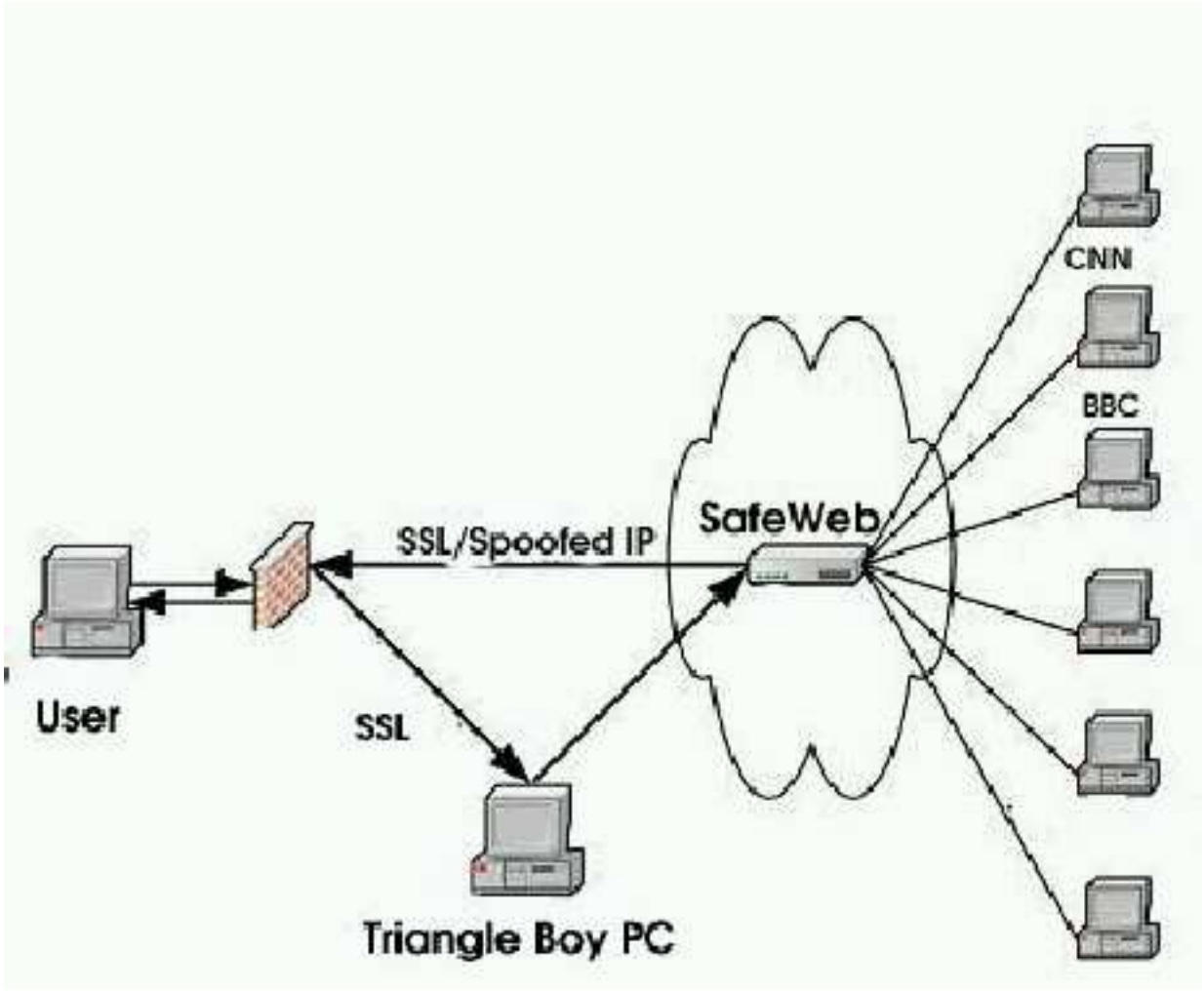
Design Goals

- Deniability for clients
  - Can’t confirm that a client is intentionally retrieving censored data
- Statistical deniability for clients
  - Browsing patterns indistinguishable from "innocent" clients
- Covertness for servers
  - Can’t discover a server that is serving censored content
  - Defense against blocking
- Communication Robustness
  - Should be difficult to disrupt request/transfer of censored content
- Performance

Anonymizing Proxies

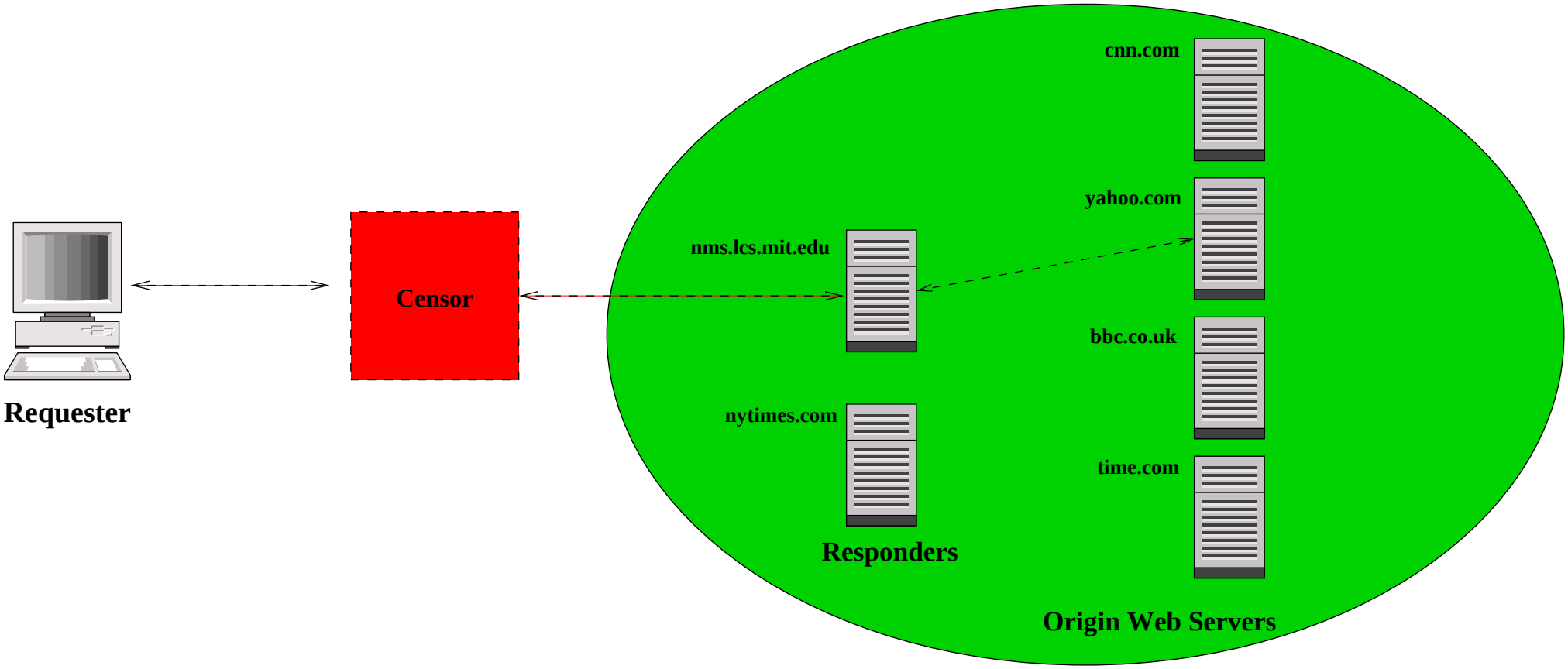


Safeweb/Triangle Boy



Circumventing censorship should be both confidential and covert!

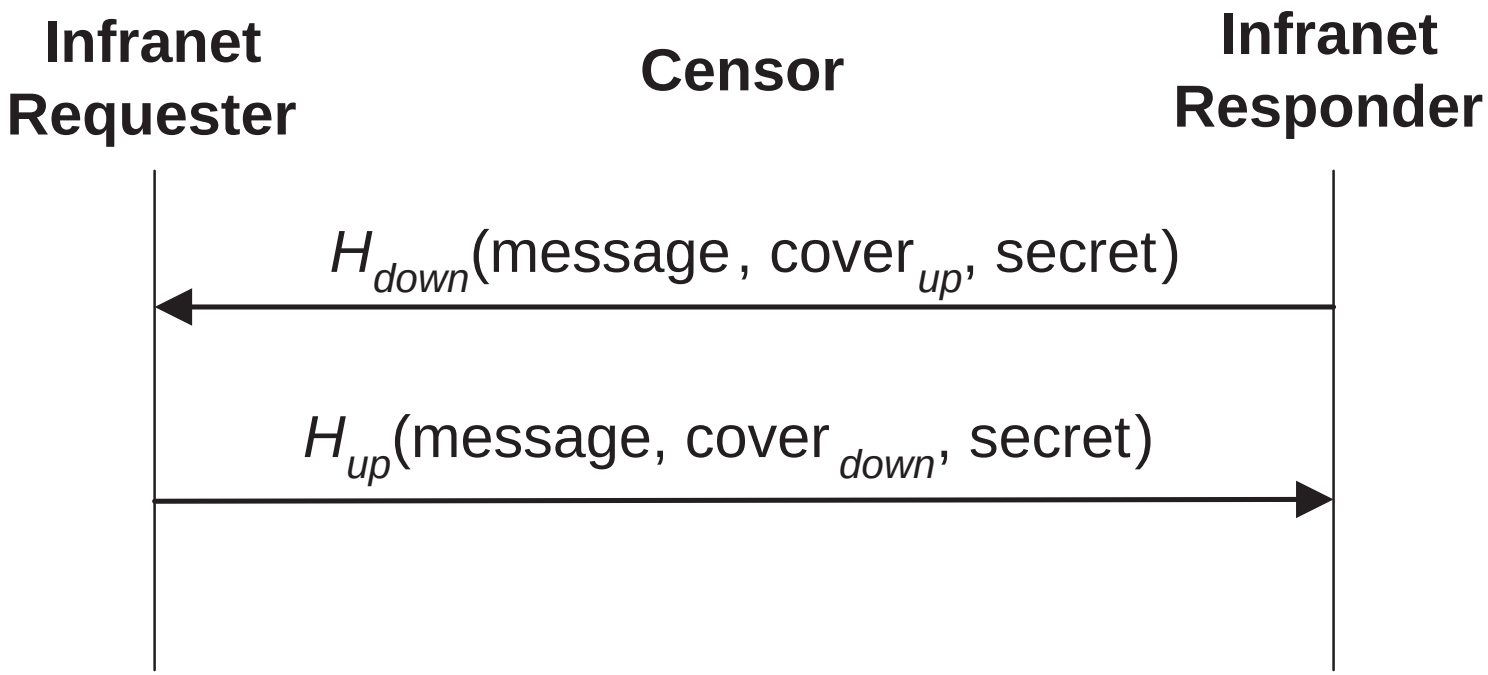
Infranet System Architecture



Communication Protocol Overview

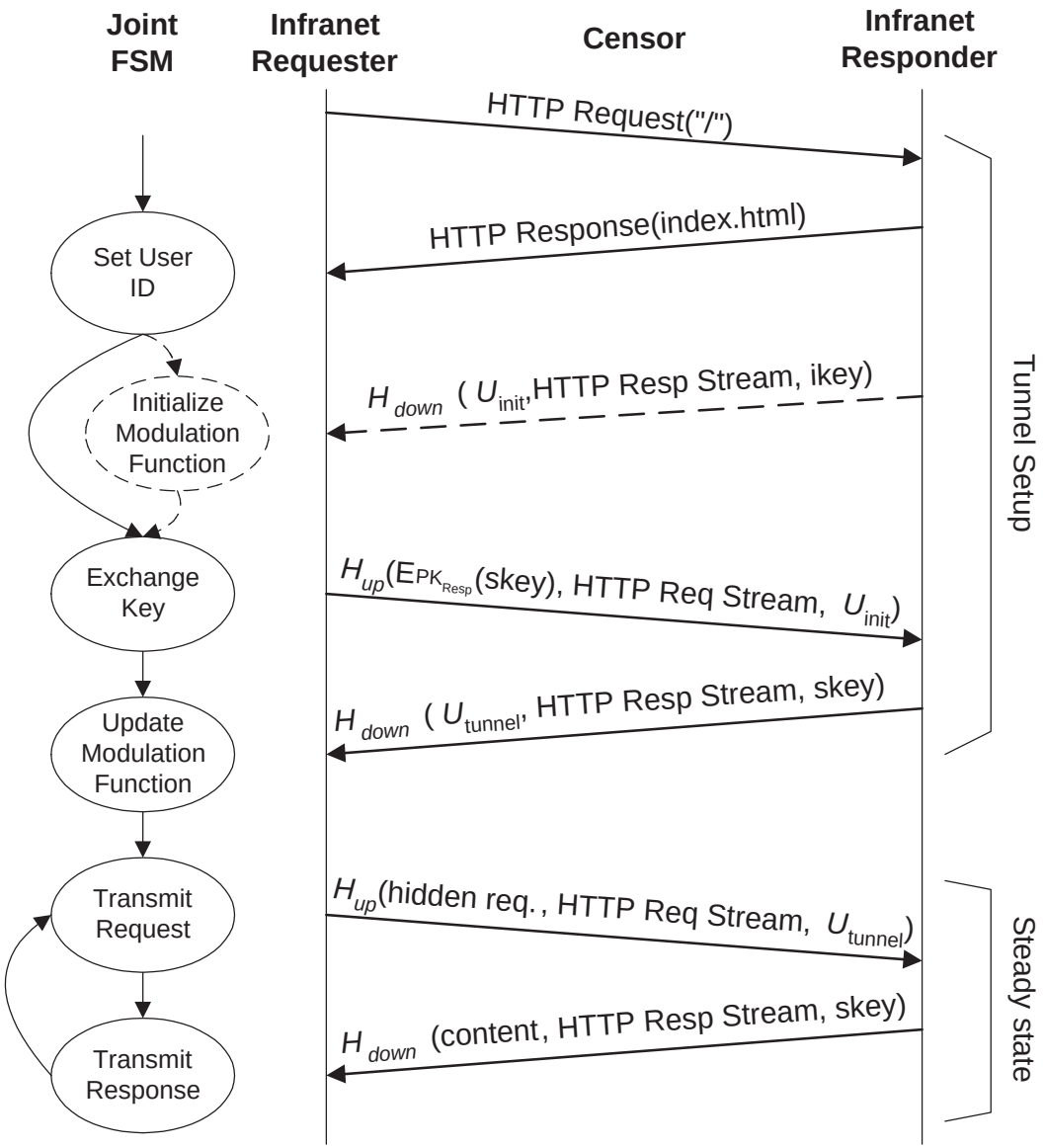
- Message exchange
  - Requests and responses are hidden in visible HTTP traffic
  - Requires a hiding function
- Symbol construction
  - Specifies the domain and range of the hiding function
  - Symbols map to message fragments
  - A group of symbols is called an alphabet
    - Upstream: set of URLs on a responder's Web site
    - Downstream: high-frequency components in images
- Modulation
  - Specifies the function itself
  - Maps a sequence of visible HTTP requests to a hidden message

Hiding

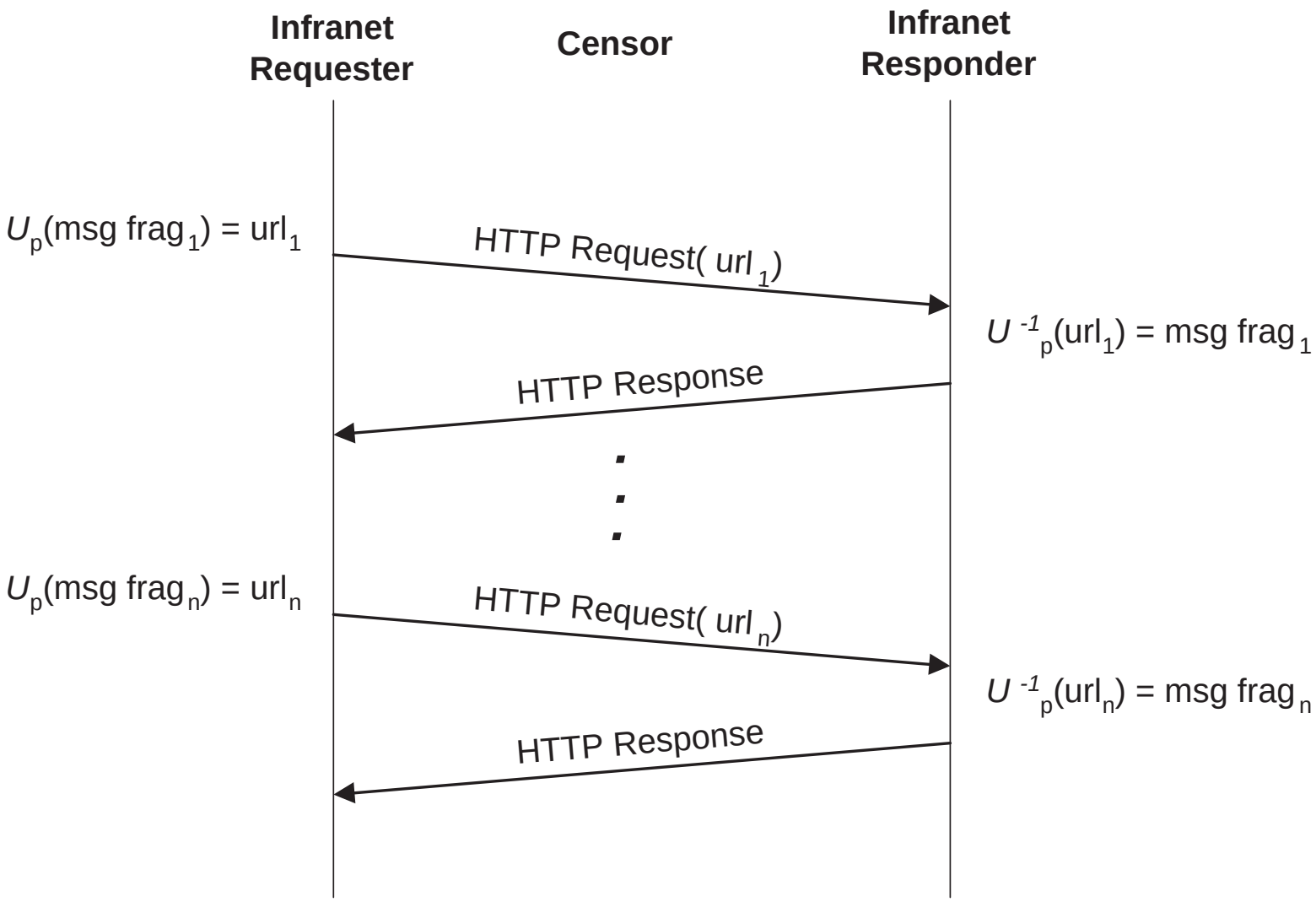


- message to hide,
- a cover medium, and
- a secret that reveals the hidden message

Tunnel Protocol



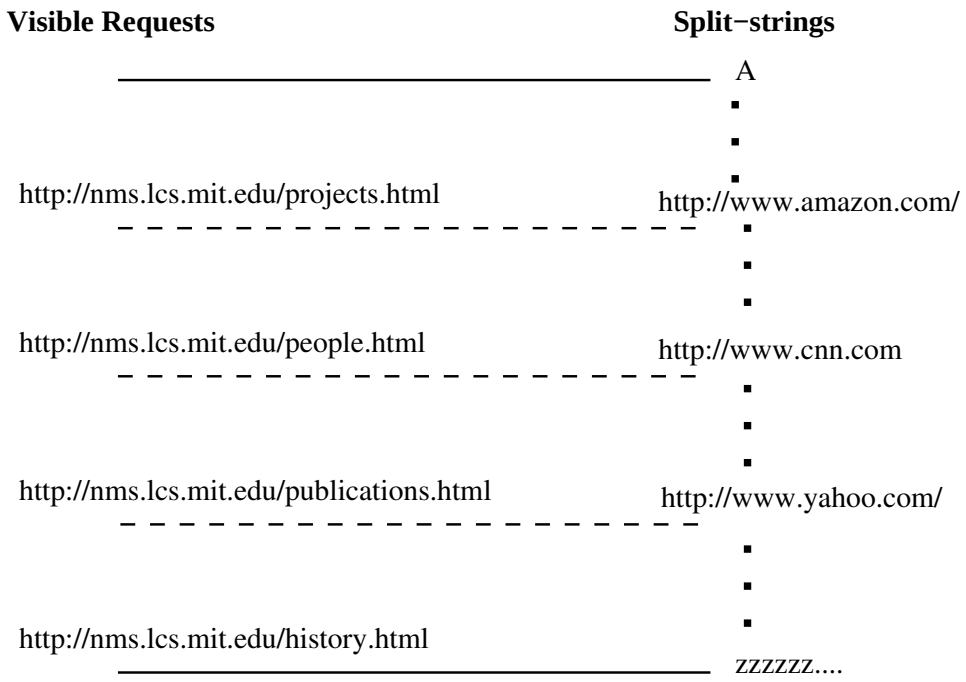
Upstream Modulation

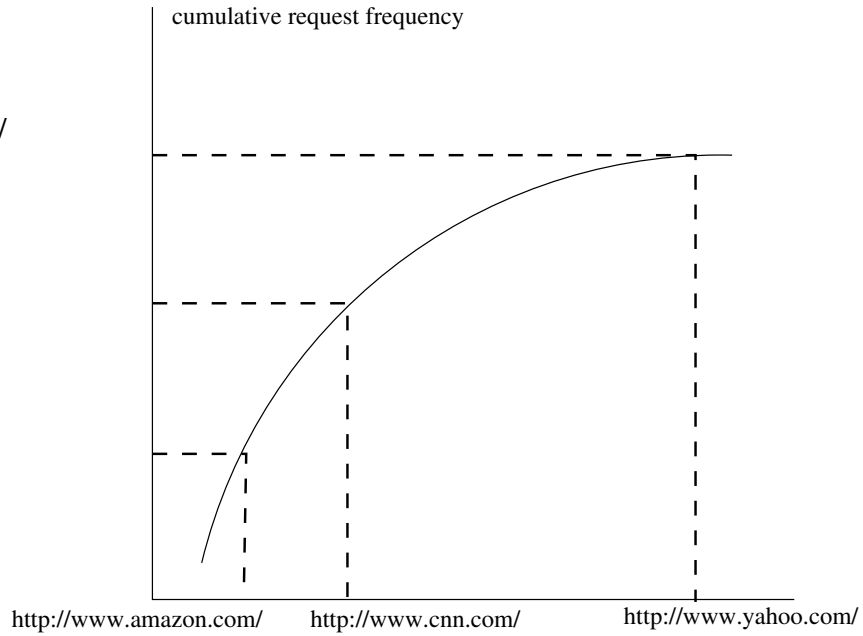


Covertness vs. Bandwidth

- Odd/Even Links
  - *Covertness:* Requester may ask for any one of half of the links at any given time
  - *Bandwidth:* 1-bit per visible HTTP request
- Links modulo k
  - *Covertness:* Requester asks for any of N/k links
  - *Bandwidth:* lg(k) bits per visible HTTP request
- Static/Dynamic Mapping
  - *Covertness:* potentially quite bad...
  - *Bandwidth:* M bits per request

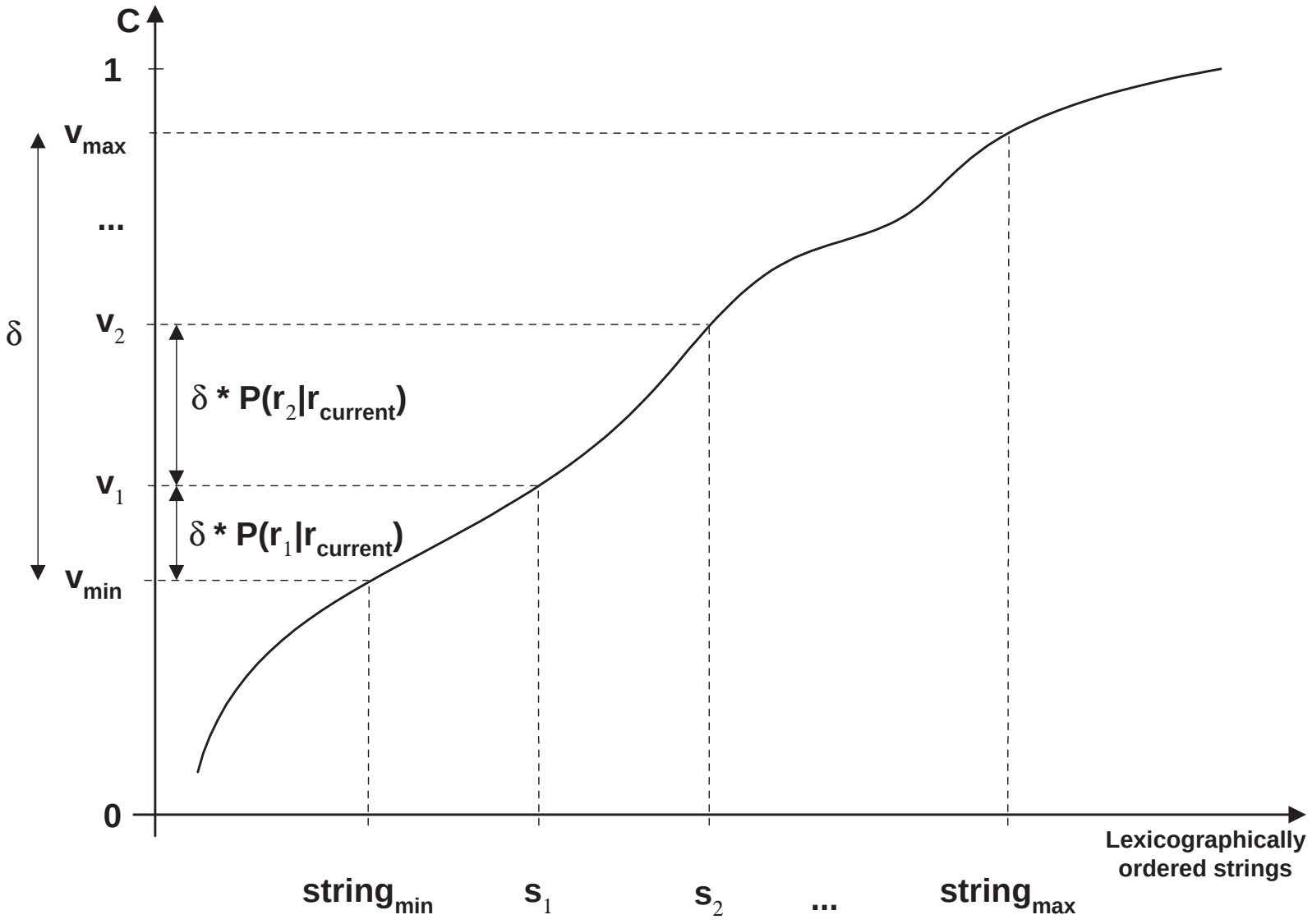
Range-Mapping





- Responder tells requester the boundaries (split-strings) for these ranges, and which visible HTTP request maps to which range.
- Requester sends a visible HTTP request to specify in which range his request lies.

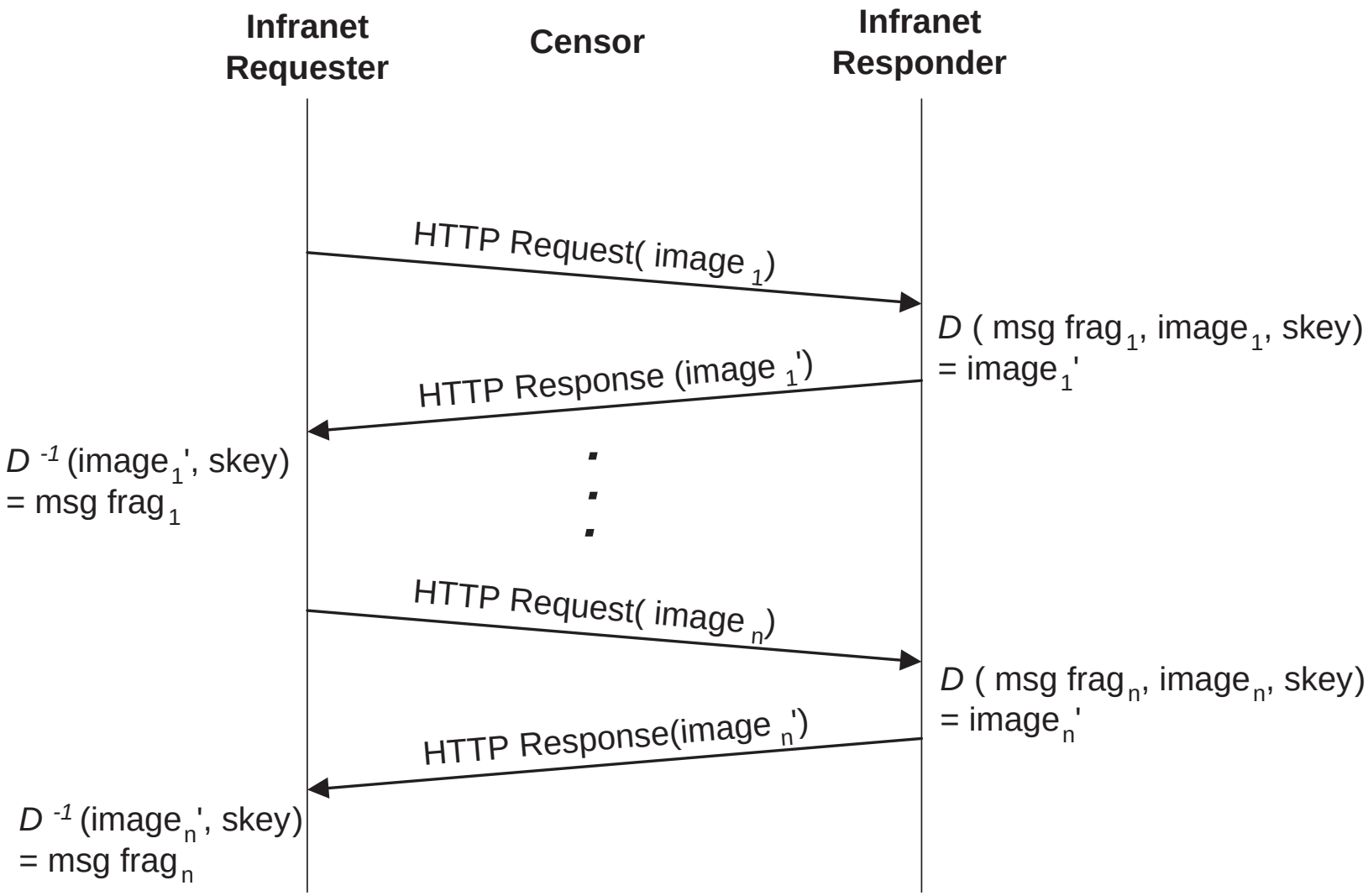
Range-Mapping



Range-Mapping

- Division according to frequently-requested censored URLs achieves good upstream bandwidth.
- Division of ranges according to conditional request probabilities achieves good covertness.
- Idea can be applied over the space of all strings (if full request is not in C).

Downstream Modulation



Steganography

- One-to-many steganography makes server covertness difficult.
  - Solution: never serve the same filename twice (helps defeat caching, too)
- Not robust. Watermarking would be better.

Security Analysis

- Discovery Attacks
  - Responders: discover and block access
  - Requesters: discover and...
  - Channel: reveal channel and its contents
- Disruptive Attacks
  - Block access to "Infranet"
  - Break tunnel protocol
  - Remove or alter:
    - hidden content
    - hidden requests

Discovery Attacks

- *Intent:* reveal requester or responder
- Passive attacks to detect unusual
  - browsing patterns
  - HTTP requests/responses

    Defense: Innocuous browsing patterns, no modification to HTTP headers themselves
- Impersonation attacks
  - censor establishes an Infranet responder and nails requesters

    Defense: later...

Disruptive Attacks

- *Intent:* disrupt communication between requester/responder
- Filtering (IP, DNS, port, etc.)

  Defense: server covertness--never return the different content for the same URL
- Request Tampering
  - reorder/remove fields from HTTP headers
  - alter downstream content

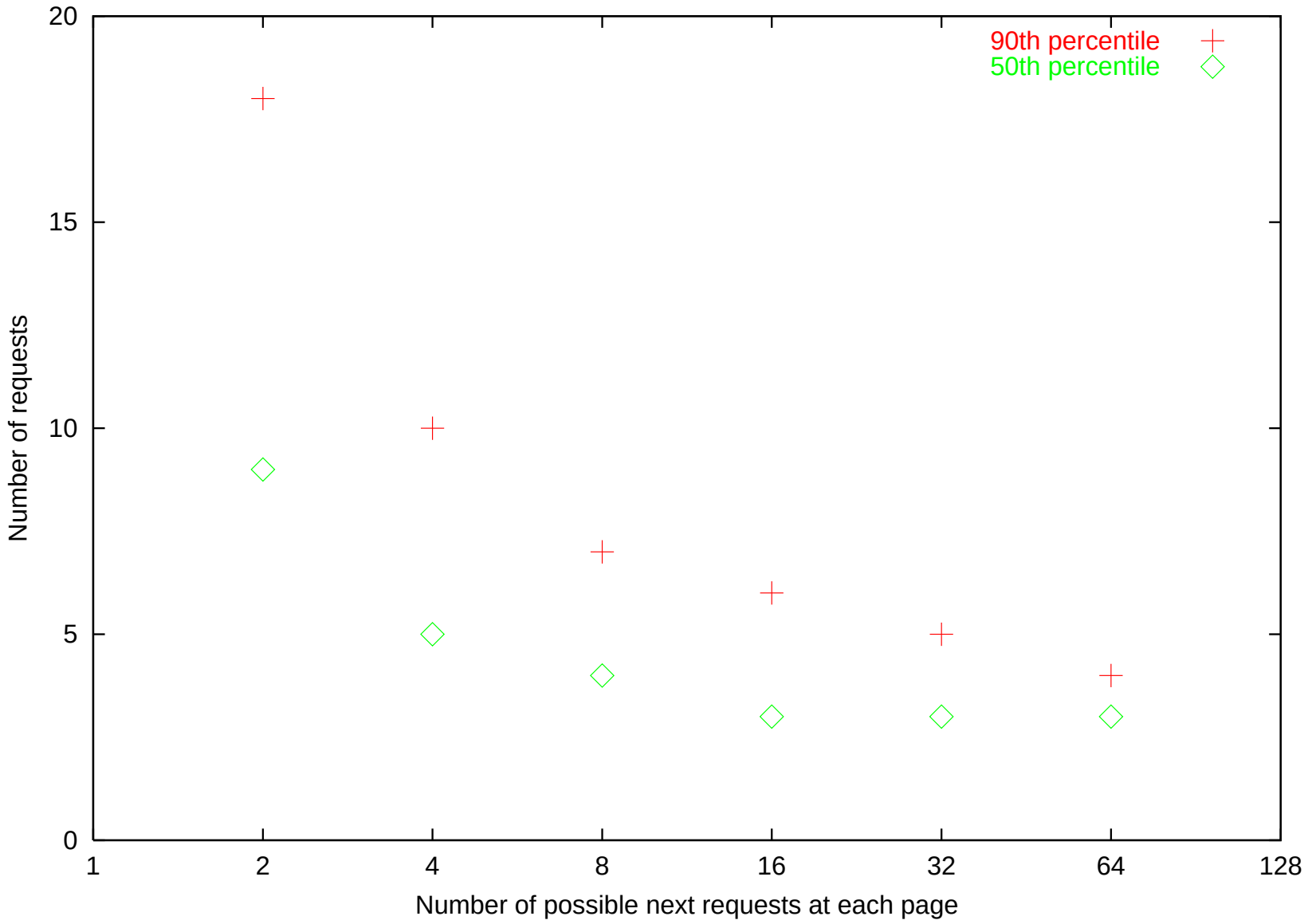
    Defense: don't hide messages in HTTP headers, no cookies, use hidden watermarks, etc.
- Session Tampering
  - insert/remove/reorder HTTP requests and responses

    Defense: error correction, headers to "echo" message to responder, etc.

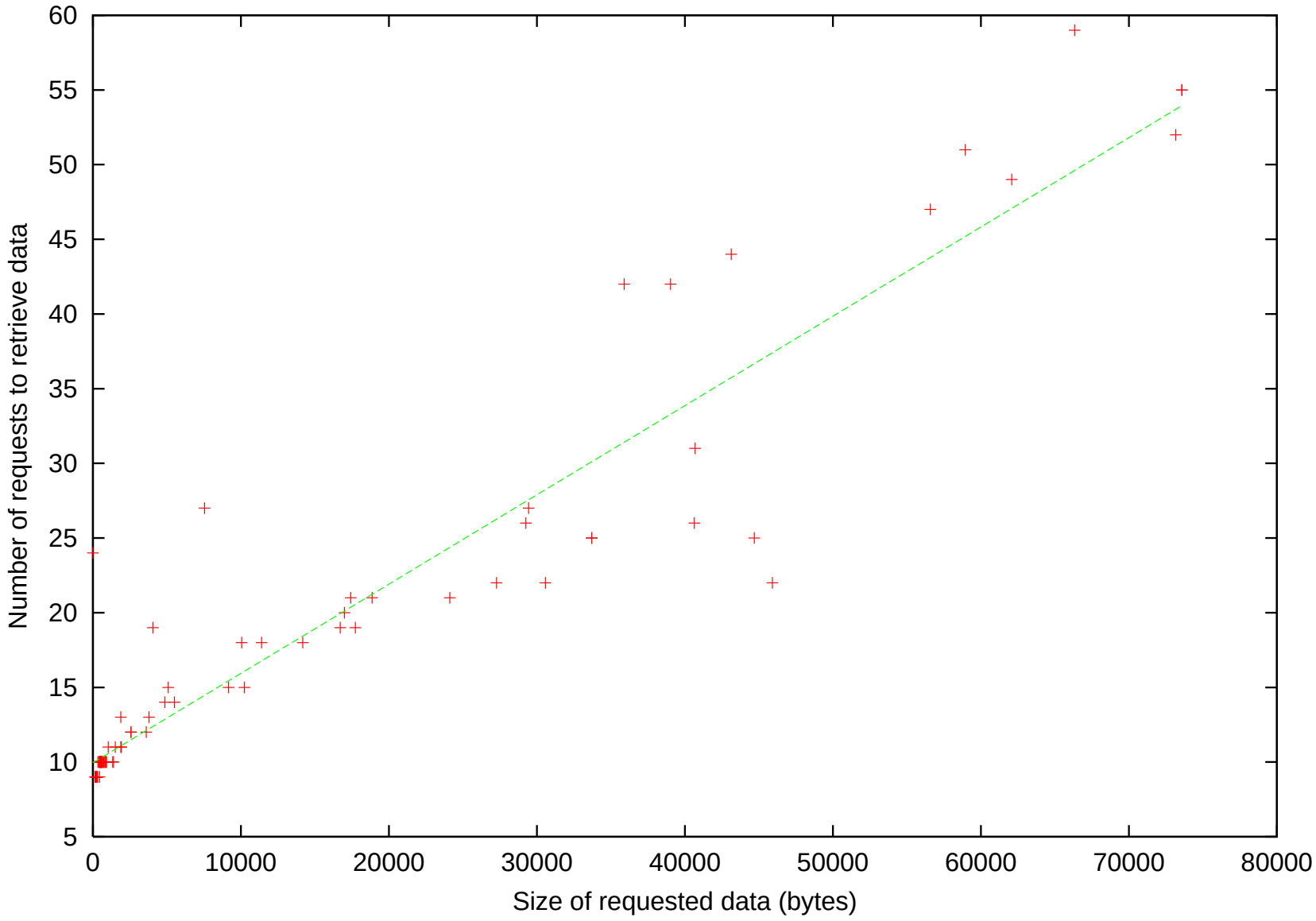
Statistical Deniability is Free

- Alphabetic coding says that we can transmit message in ceil(H(C)) + 2 symbols
- We’re sending lg(k) bits per symbol
- Can send message in ceil(H(C)/lg(k)) + 2 visible HTTP requests
- Probability of sending a symbol is the probability that the hidden request lies in that range

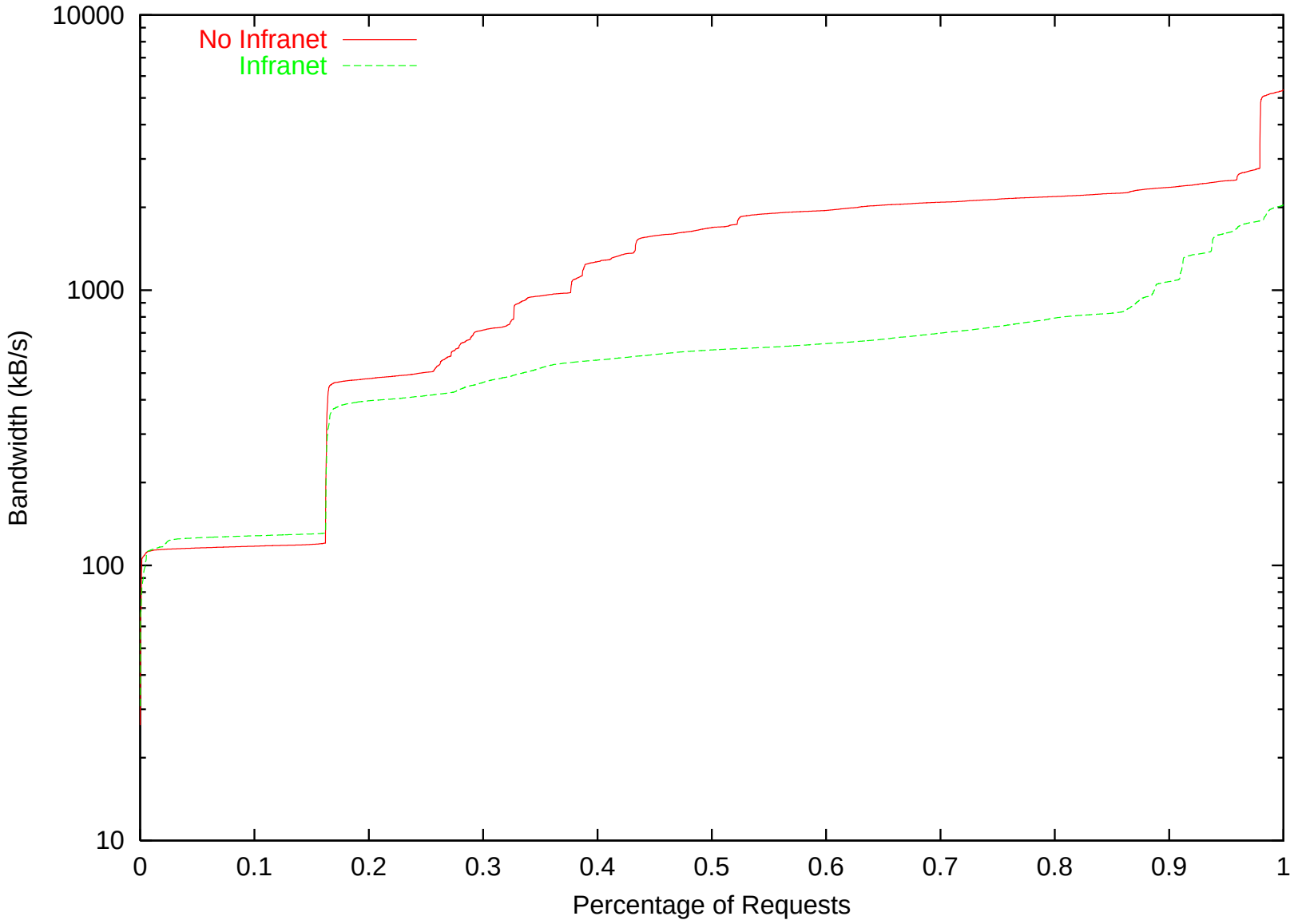
More Visible Links => Greater Upstream Bandwidth



Downstream Performance



Server Overhead

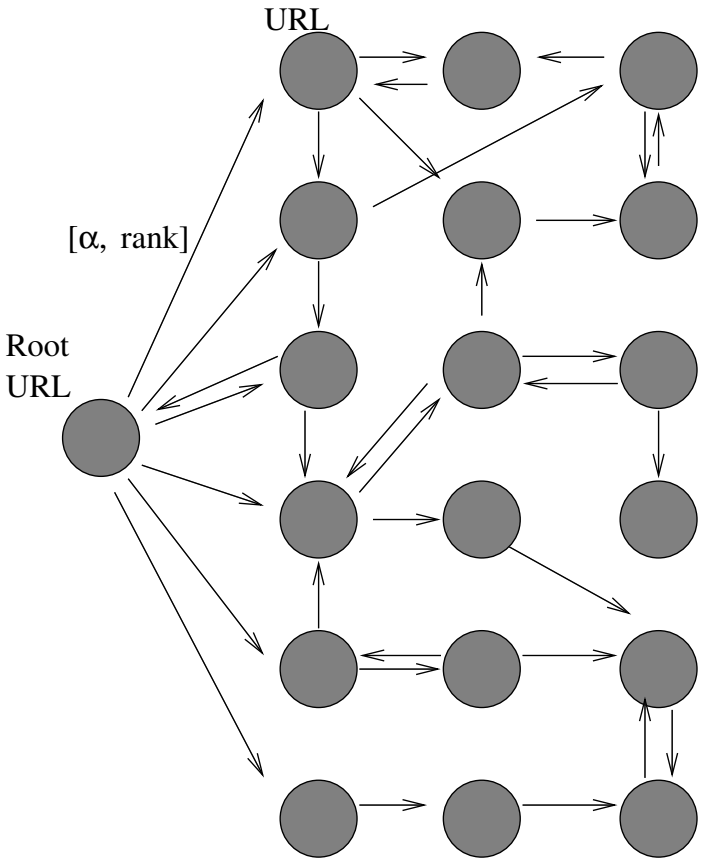


Demo

Current Work

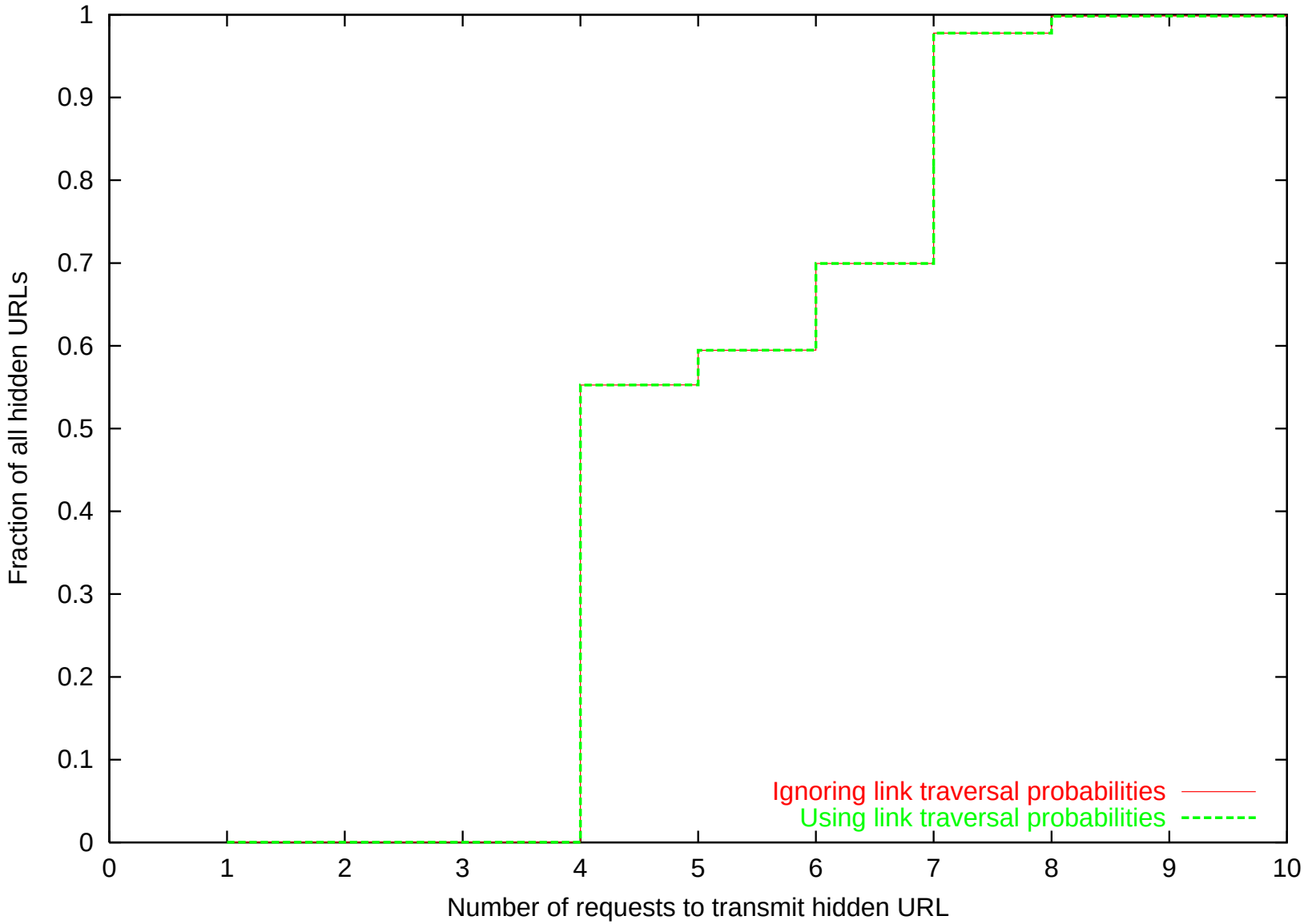
- "Bootstrapping"
- Allow for untrusted responders
- Cookieless Implementation
- More robust downstream modulation
- Versioning

Digraph Dictionary



- Nodes map to visible HTTP requests
- Edges map to message fragments
- Graph corresponds to user browsing patterns

Range-Mapping

- Scenario
  - Responder knows some frequency distribution on commonly-requested hidden content, C
  - Requester wants to send some message in C
  - From any given page, there are k possible next visible HTTP requests
- Approach
  - Divide C into k ranges, with the size of each range corresponding to the conditional probability that request is made.
  - Responder tells requester the boundaries (split-strings) for these ranges, and which visible HTTP request maps to which range.
  - Requester sends a visible HTTP request to specify in which range his request lies.

Statistical Deniability is Free