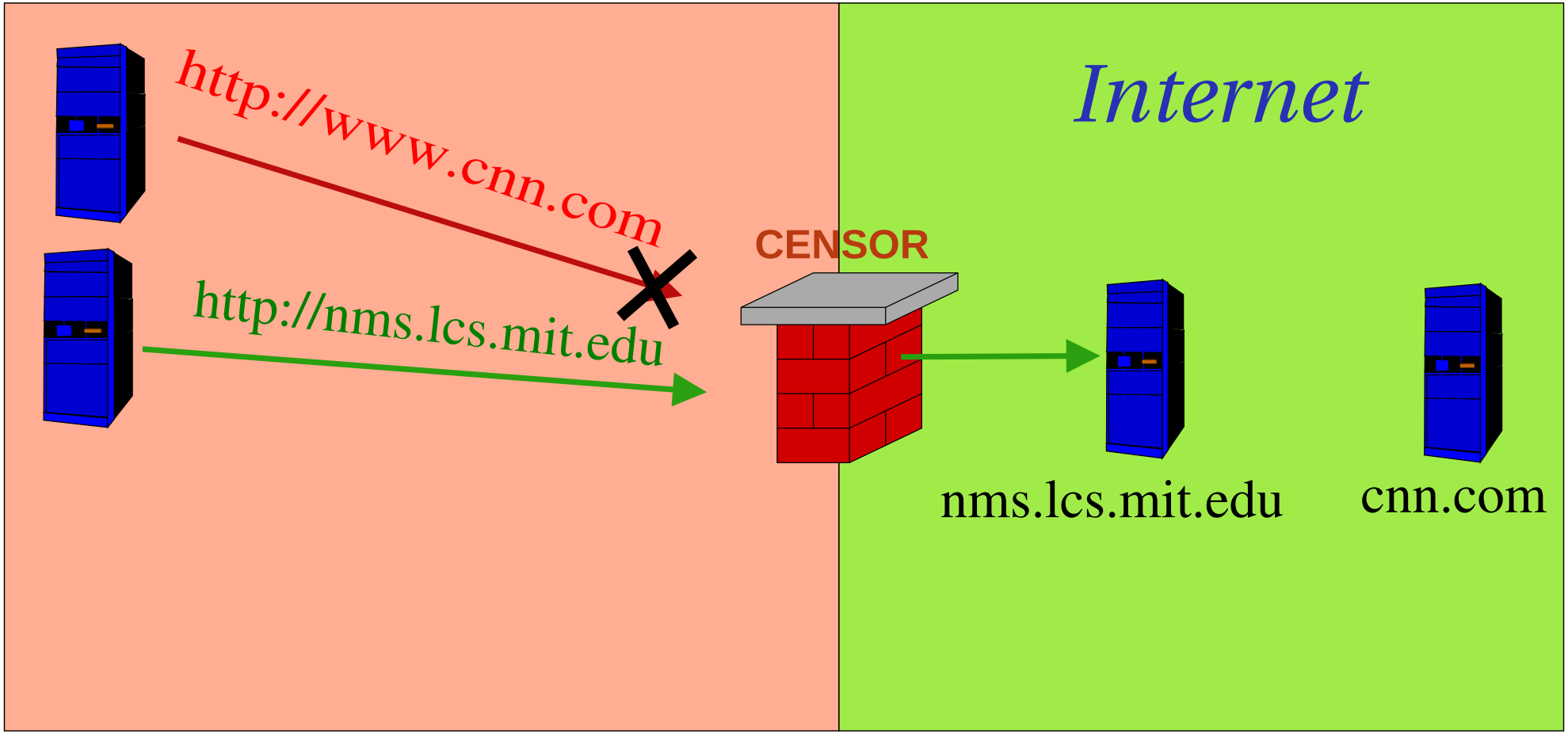
Infranet: Circumventing Web Censorship and Surveillance

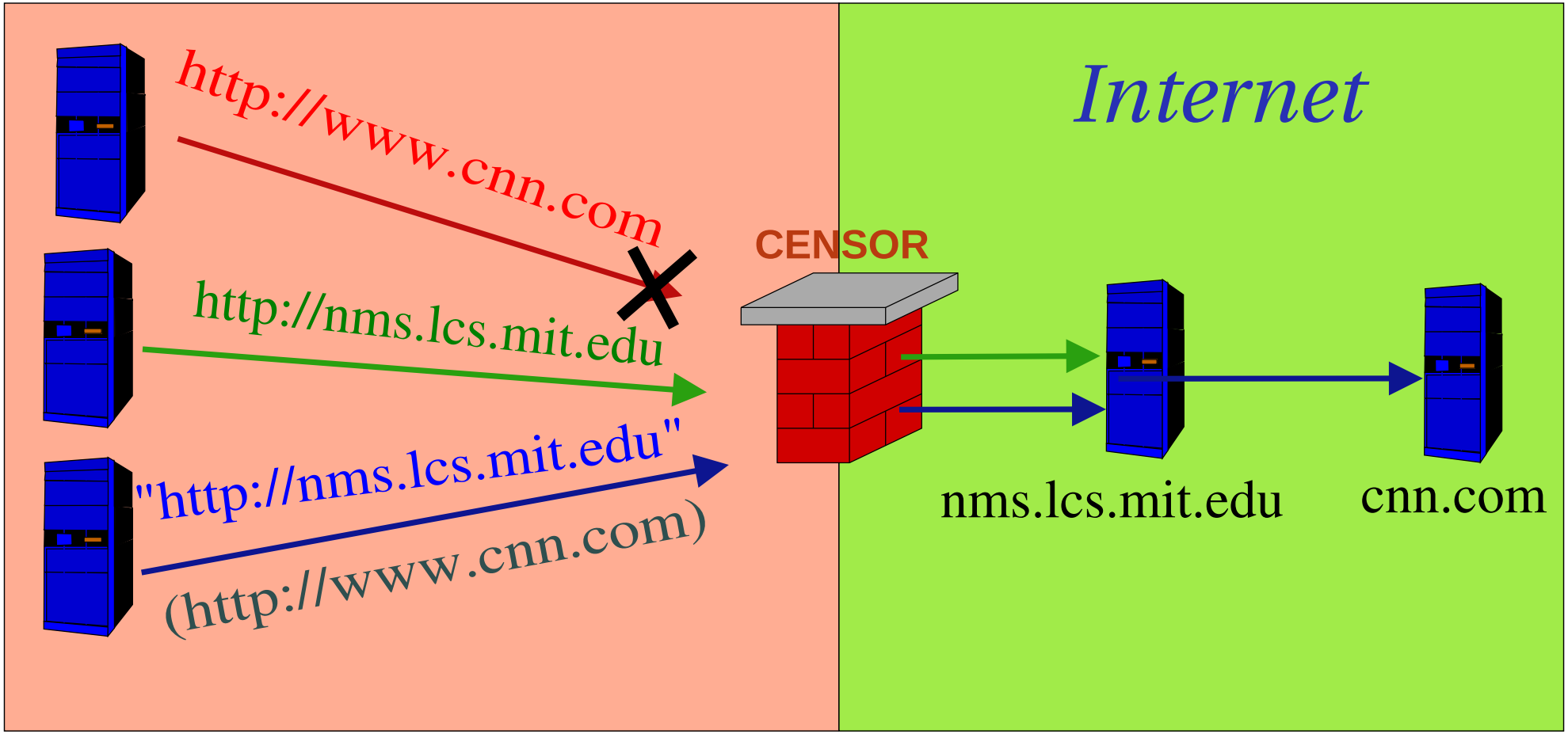
Nick Feamster, Magdalena Balazinska,
Greg Harfst, Hari Balakrishnan, David Karger
M.I.T. Laboratory for Computer Science

http://nms.lcs.mit.edu/infranet/

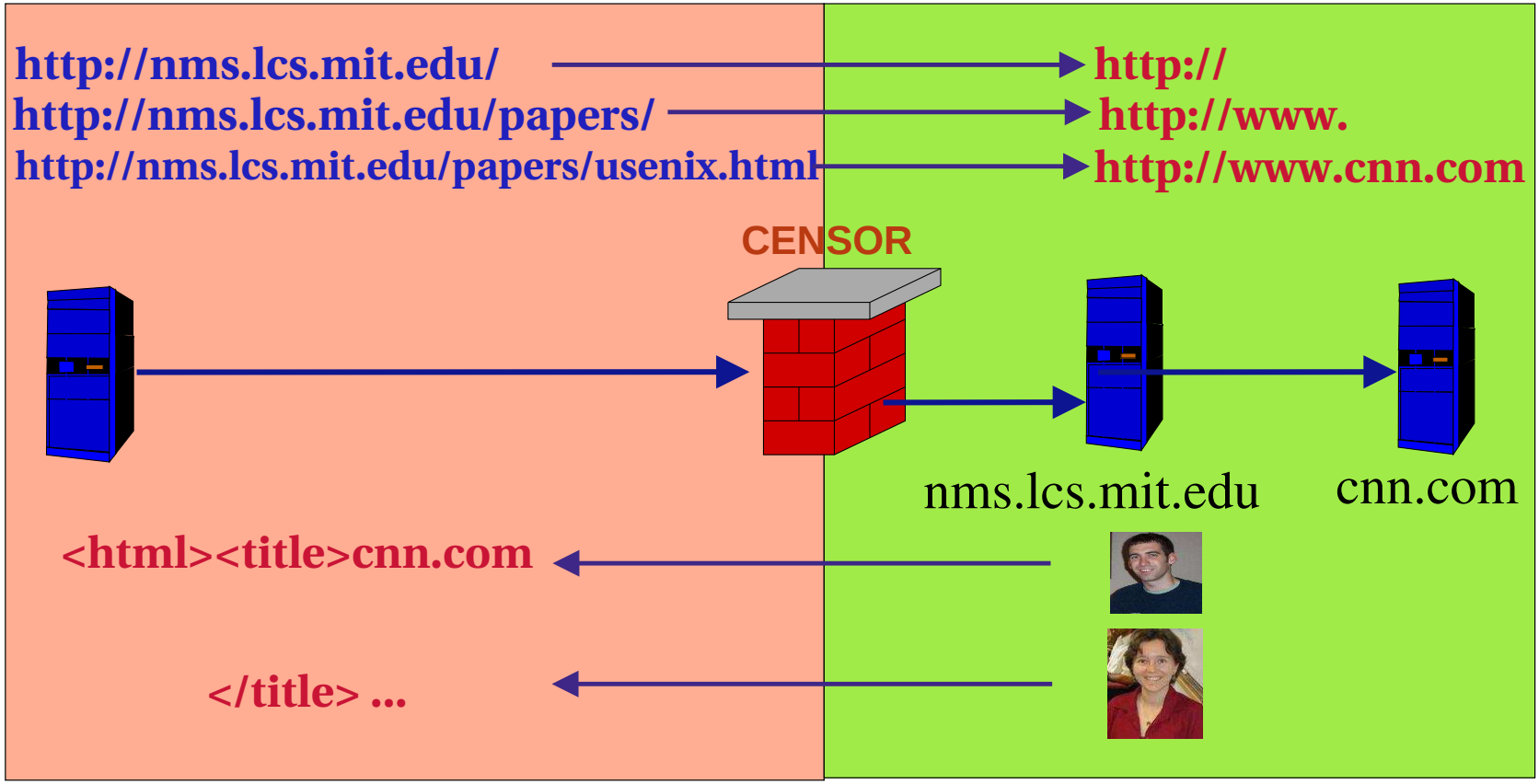
The Big Picture



The Big Picture



How Infranet Works



- Use Infranet requester proxy (on localhost)
- Upstream request in sequence of HTTP requests
- Downstream response in images

Censor

- Restrictive government, corporate firewall, etc.
- Discovery Attacks: Notice unusual-looking Web traffic.
  - monitors Web access for "inappropriate use"
  - watch Web traffic for inappropriate access attempts
  - *watch for suspicious looking Web access patterns*
  - *watch for use of circumvention software*
  - watch for suspicious inter-request times
- Disruptive Attacks: Keep the endpoints from talking.
  - blocks access to certain Web sites
  - *attempts to block access to circumvention software (e.g., blocking SSL, disrupting communication, etc.)*

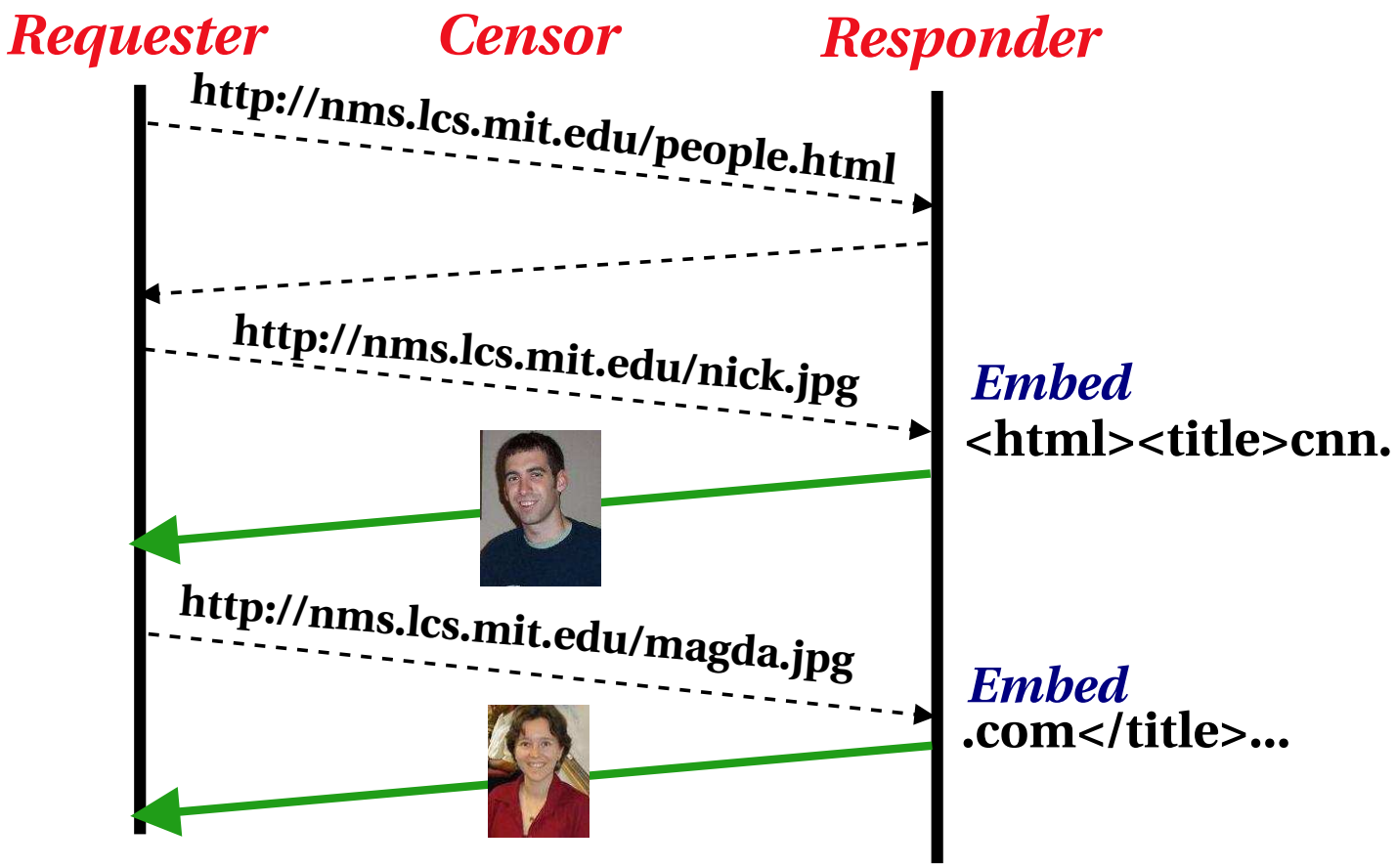
Design Goals

- *Deniability for clients*
  - Can’t confirm that a client is intentionally retrieving censored data
- *Statistical deniability for clients*
  - Web traffic doesn’t look unusual
- *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
- *Reasonable Performance*

Related Systems: Triangle Boy, Peekabooty, etc.

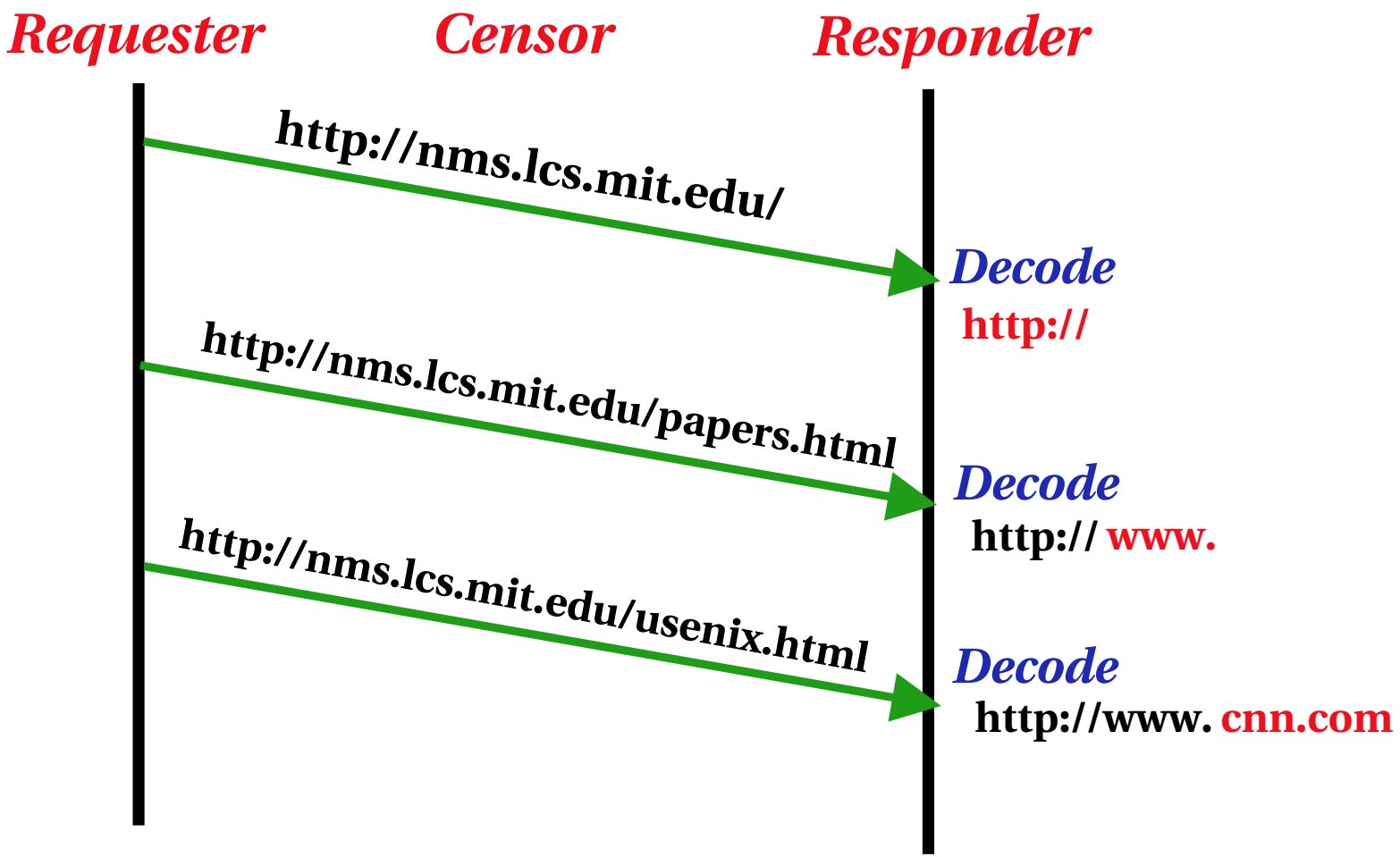
- *Deniability for clients*
  - Existing systems rely on SSL, vulnerable to fingerprinting
- *Statistical deniability for clients*
  - SSL traffic looks suspicious
  - No attempt to conceal suspicious traffic patterns
- *Covertness for servers*
  - Servers make no attempt to conceal their existence
  - Suspicious traffic patterns may result in discovery and blocking
- *Communication Robustness*
  - SSL can be blocked (e.g., unsigned server certificates)

Downstream Communication ("Downloading")



- Embed data in images, recover by shared secret
- Steganography is not ideal: can't reuse cover image
- Web cams are wonderful.

Upstream Communication ("Requesting")



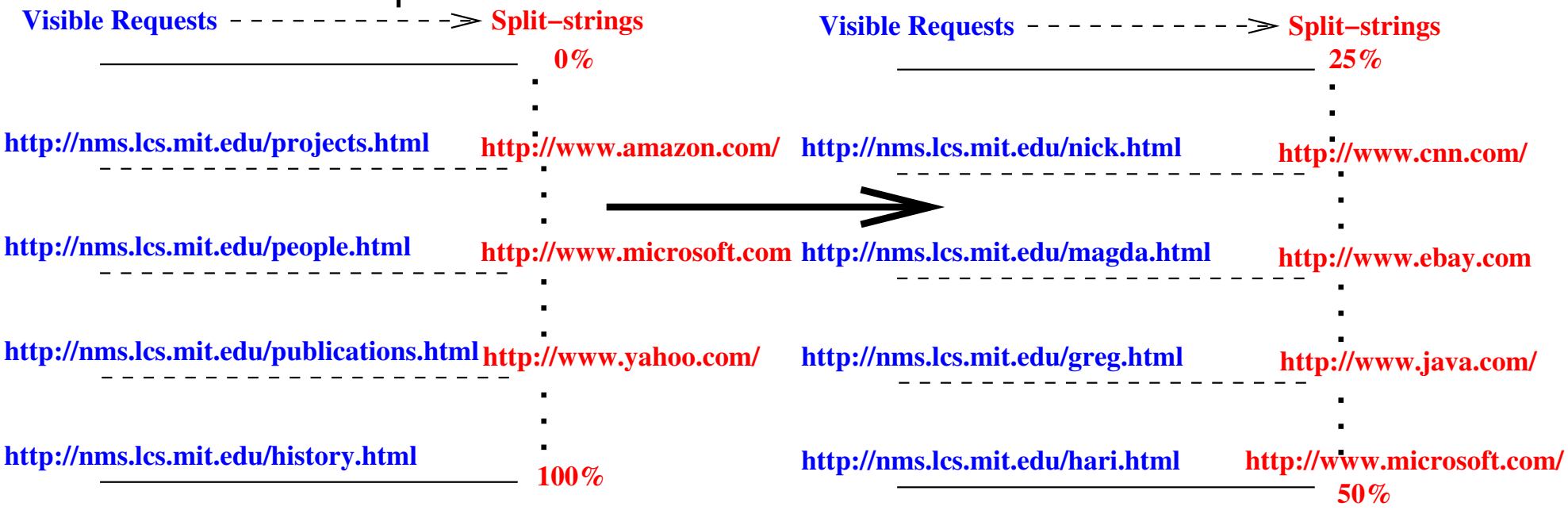
- Hidden message => sequence of HTTP requests
- Mapping function: secret, critical to deniability

Simple Schemes: Covertness/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 Mapping
  - *Covertness:* potentially quite bad...
  - *Bandwidth:* M bits per request

Range-Mapping: Web Surfing, 20 Questions-Style

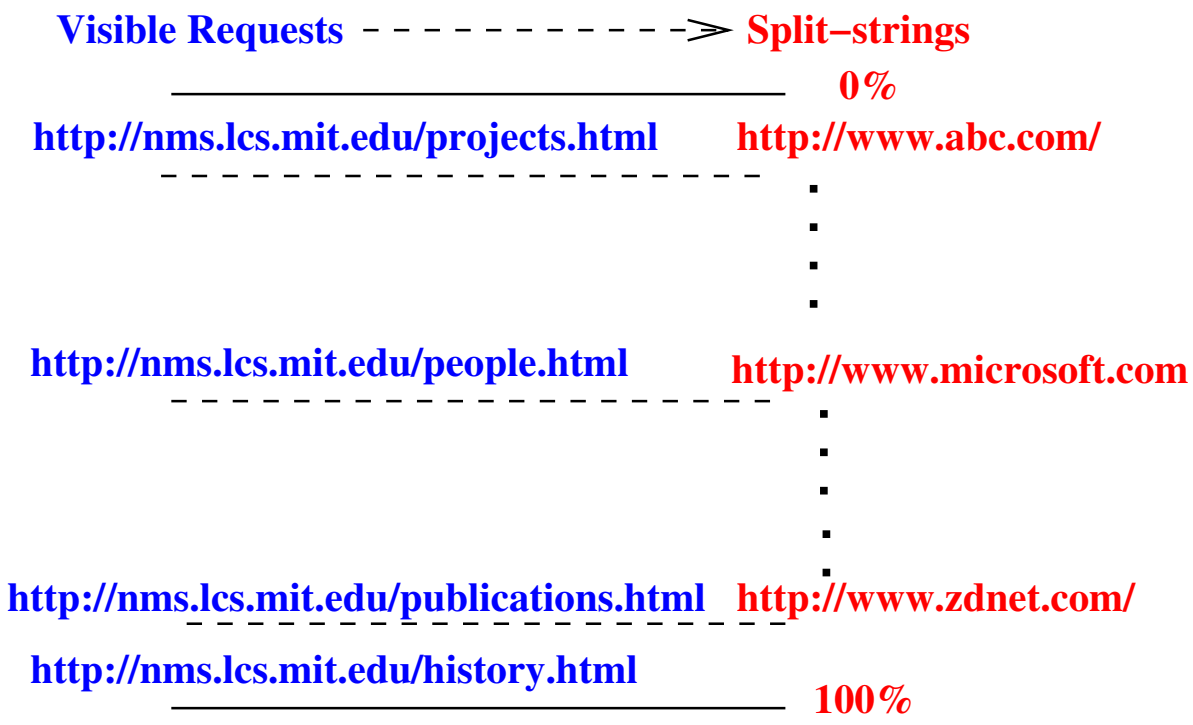
- Assume: Some set of censored URLs are commonly requested
- Responder tells requester
  - the boundaries (split-strings) for ranges in this set, and
  - the mapping between visible HTTP requests and split-strings
- Requester tells responder
  - a visible HTTP request



but...not all requests are equally likely!

Getting Statistical Deniability

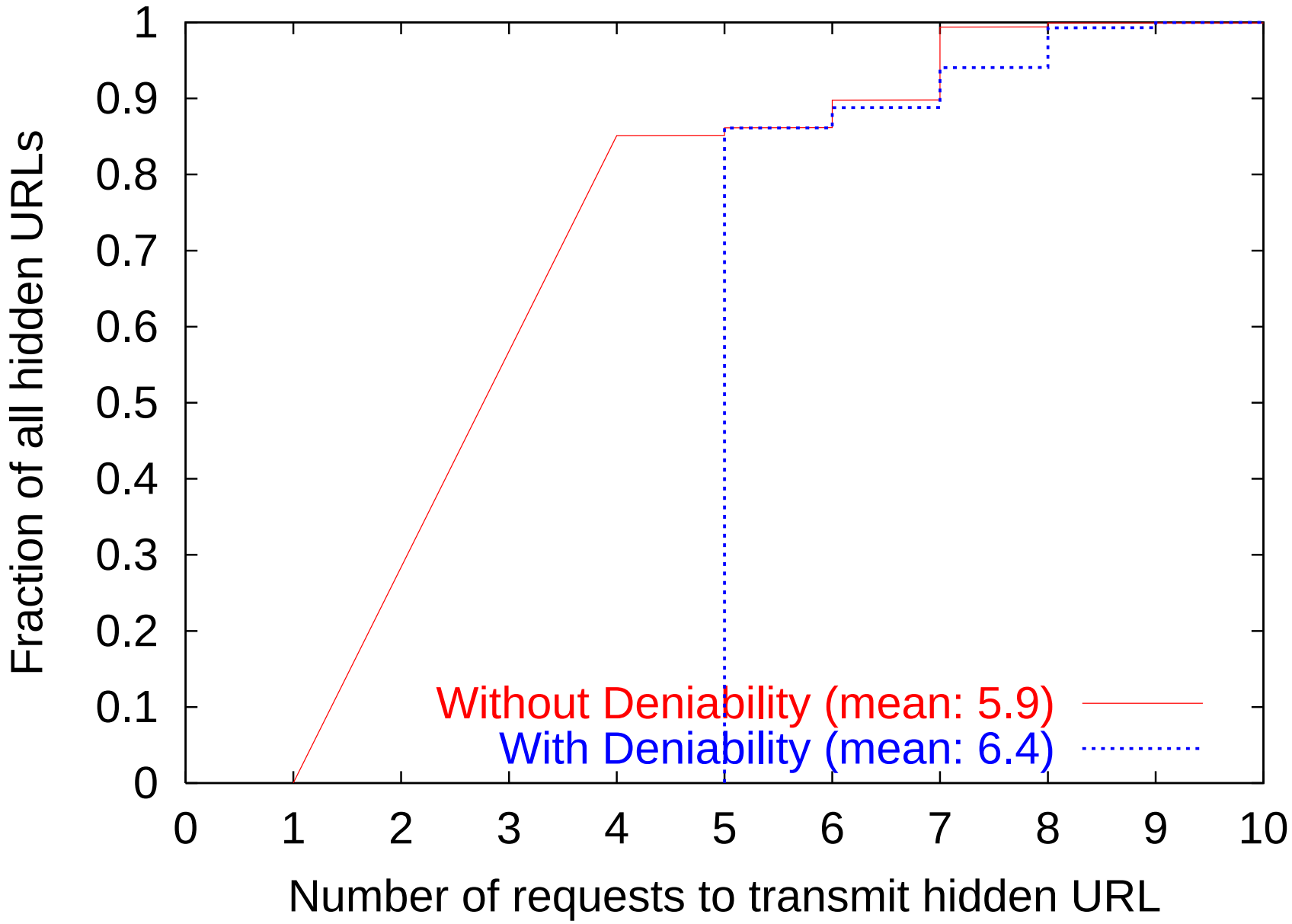
- Divide the corpus according to more likely visible HTTP requests.
- Arithmetic coding says that our expected number of requests is the same!



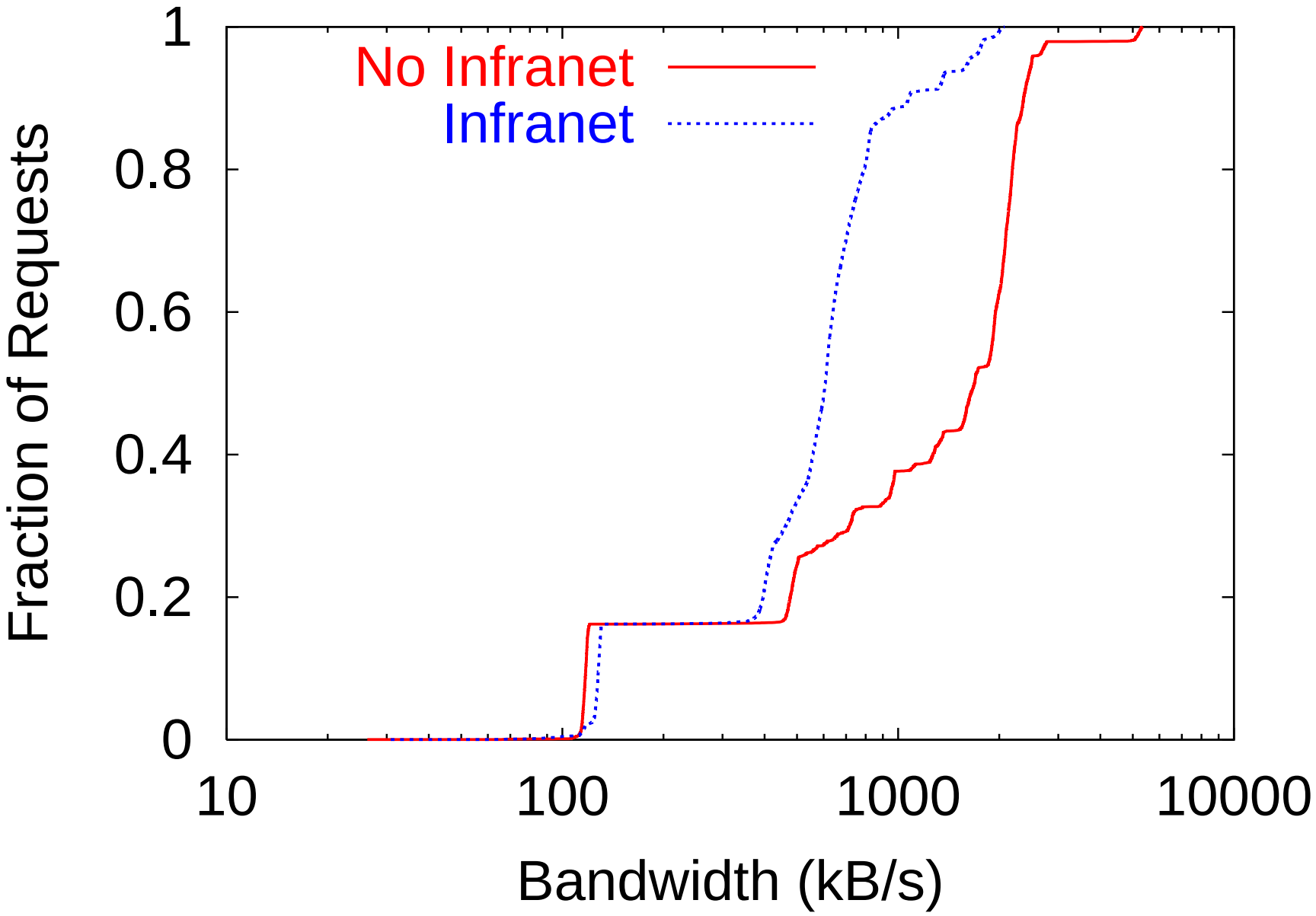
Range-Mapping

- Search through set of frequently-requested censored URLs achieves good upstream bandwidth.
- Division of ranges according to conditional request probabilities achieves deniability and covertness.
- Idea can be applied over the space of all strings.

Statistical Deniability is Free



Server Covertness is Not Free



Security Analysis

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

Discovery Attacks

- *Intent:* reveal requester or responder
- Passive attacks to detect unusual
  - browsing patterns (including request timing)
  - HTTP requests/responses

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

    Defense: separate forwarding and decoding of symbols

Disruptive Attacks

- *Intent:* disrupt communication between requester and 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.

Conclusion

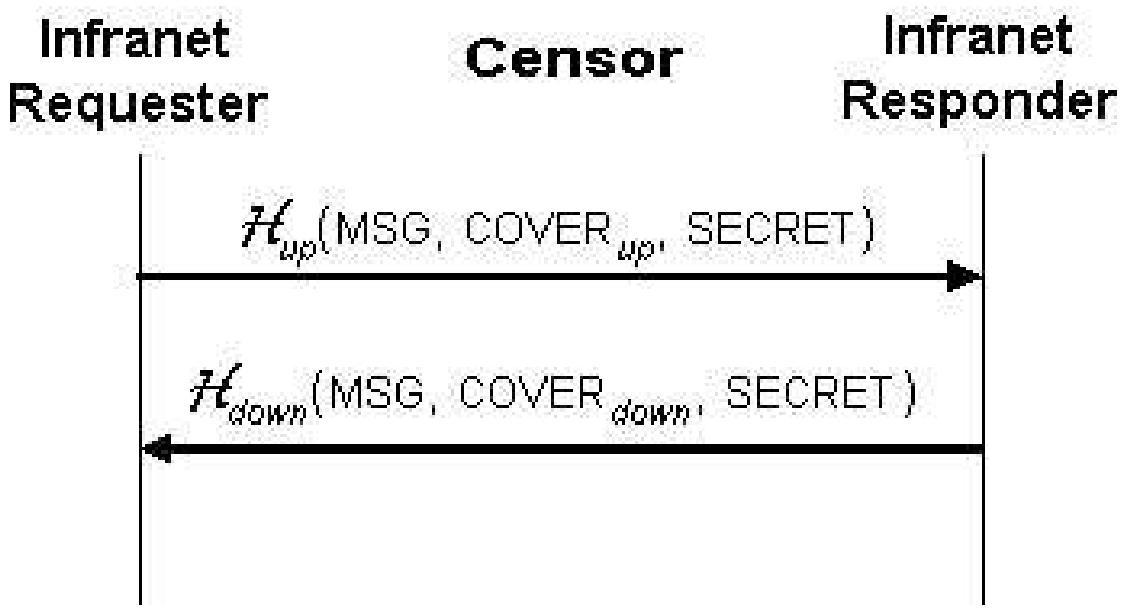
- Infranet hides censored requests and responses in innocuous-looking HTTP request/response streams
  - client deniability
  - server covertness
  - reasonable robustness
- Future work
  - robustness
  - software distribution
  - server discovery

http://nms.lcs.mit.edu/infranet/

Communication Protocol Overview

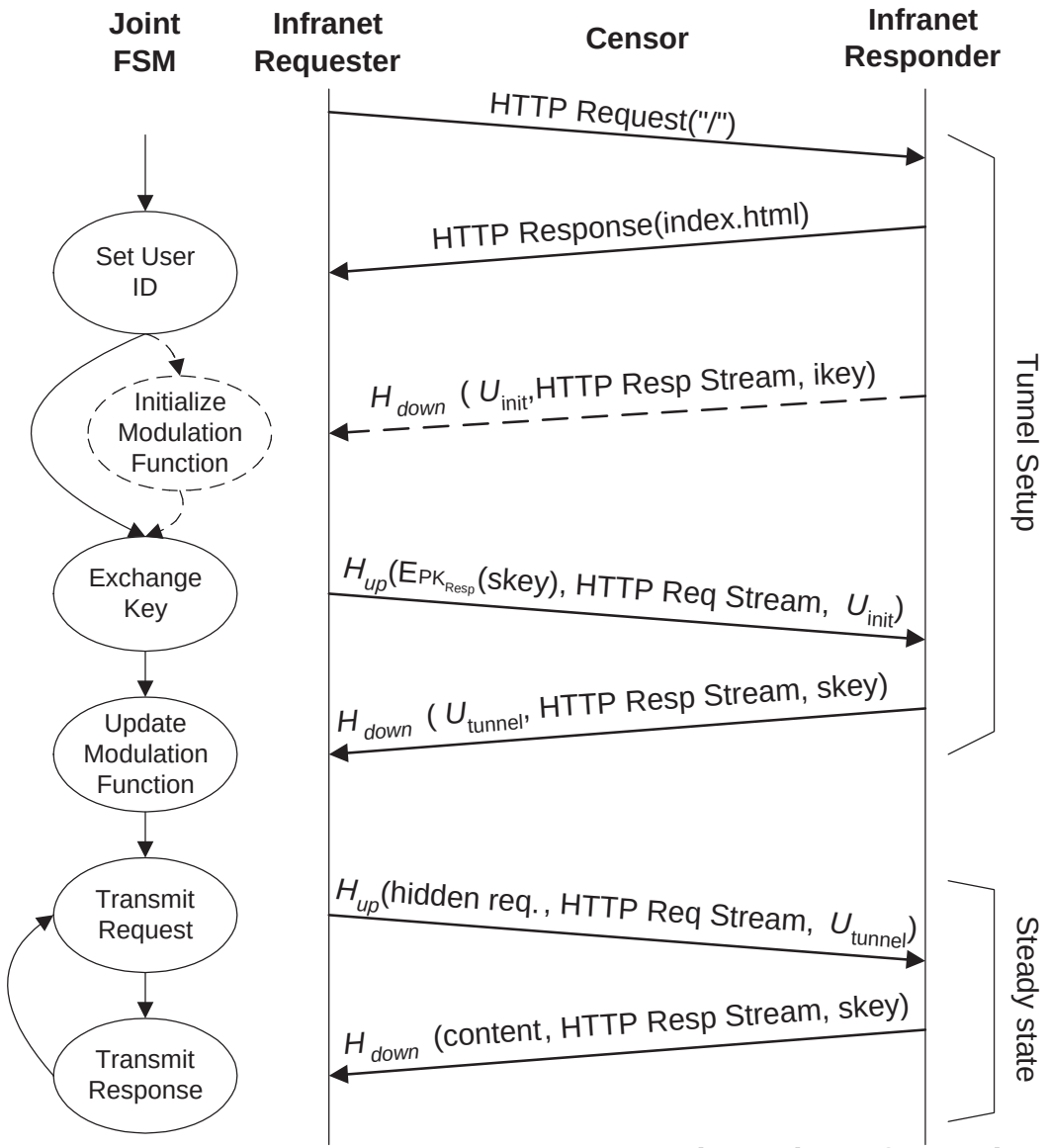
- Message exchange
  - Requests and responses are hidden in visible HTTP traffic
  - Requires a hiding function
  - Messages exchanged as a stream of symbols
- Symbol construction
  - How symbols map to message fragments
  - A group of symbols is called an alphabet
    - Upstream: set of URLs on a responder's Web site
    - Downstream: high-frequency components in images
- Modulation
  - Specifies a particular function for symbol construction
  - Maps a sequence of visible HTTP requests to a hidden message

Hiding



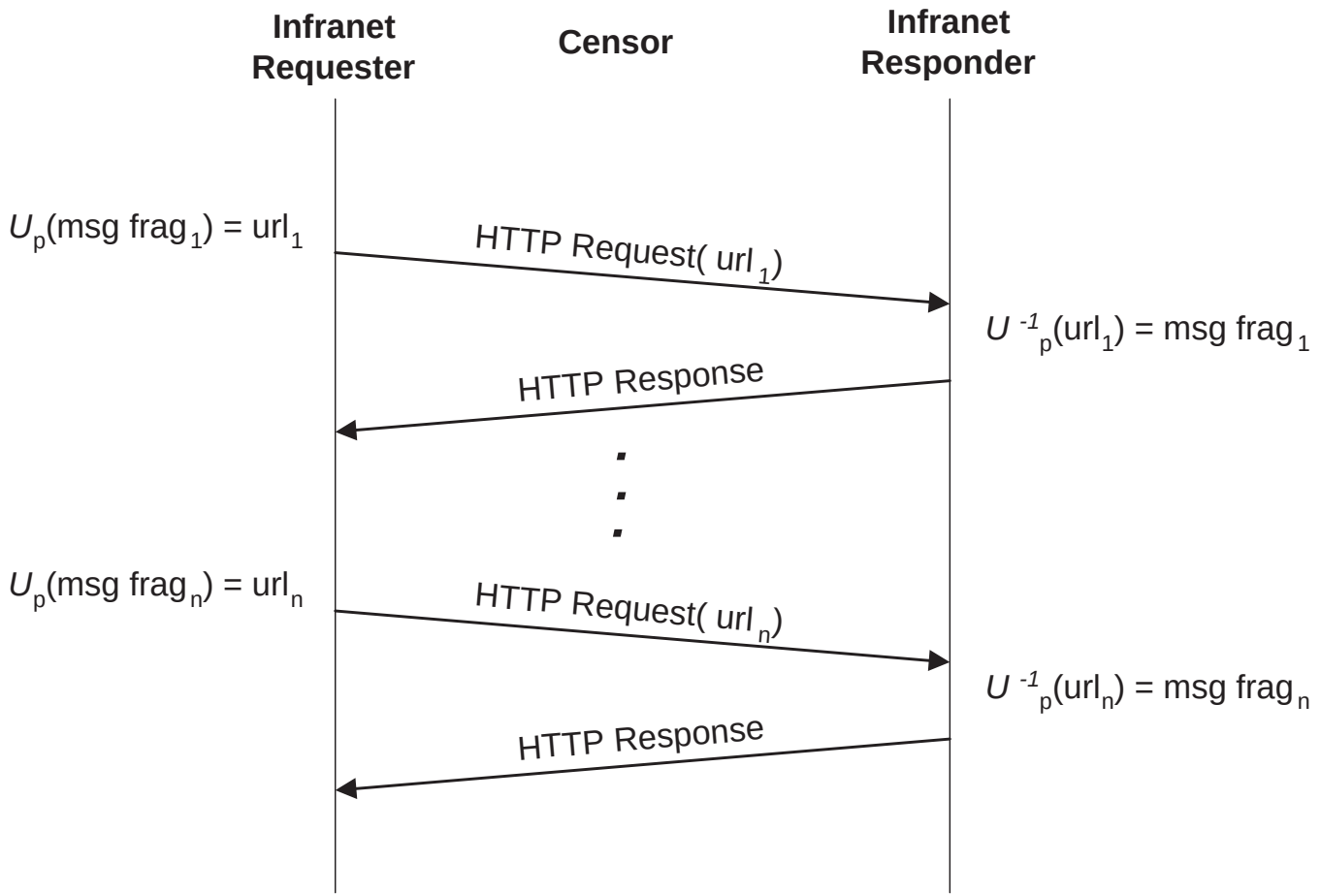
- Requires:
  - a message to hide,
  - a cover medium, and
  - a secret that reveals the hidden message
- Without knowledge of secret, can't reveal message (or even prove that a message is being hidden)
- Secret exchange...

Tunnel Protocol



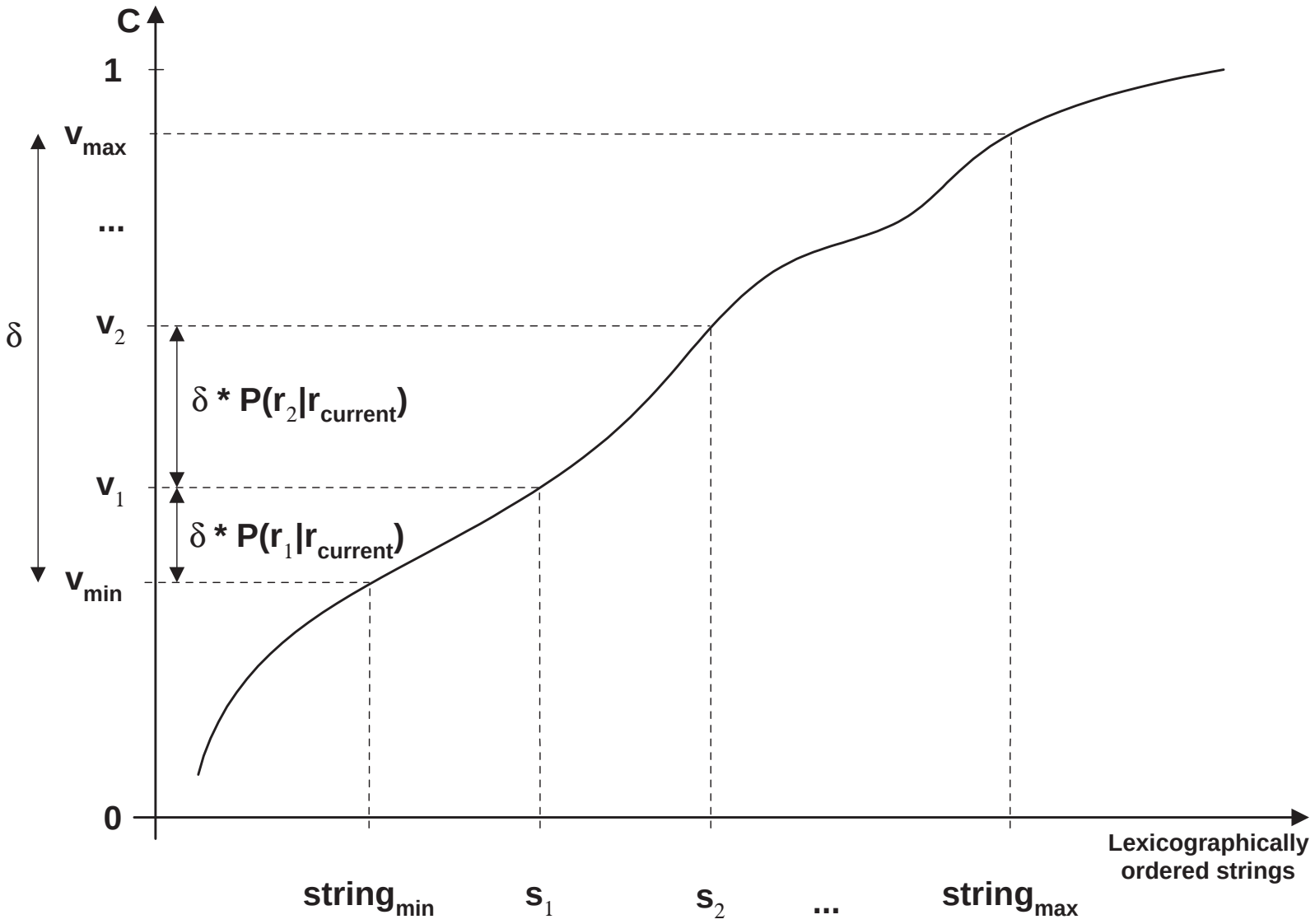
Requester learns upstream modulation function.
Responder learns secret key for downstream hiding.

Upstream Modulation

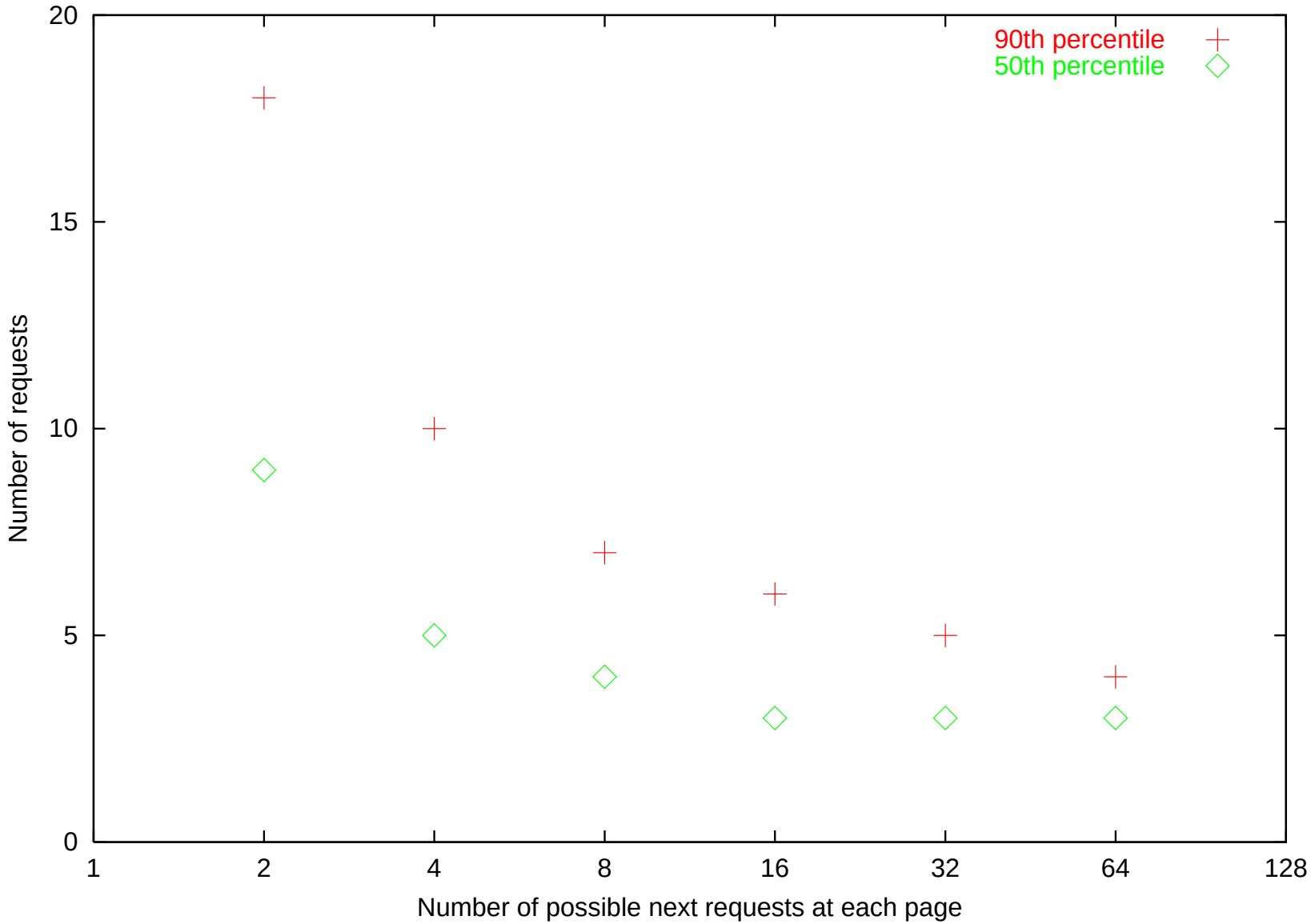


- Hidden message => sequence of HTTP requests
- Tunnel modulation function is secret
- Choice of modulation function is important!

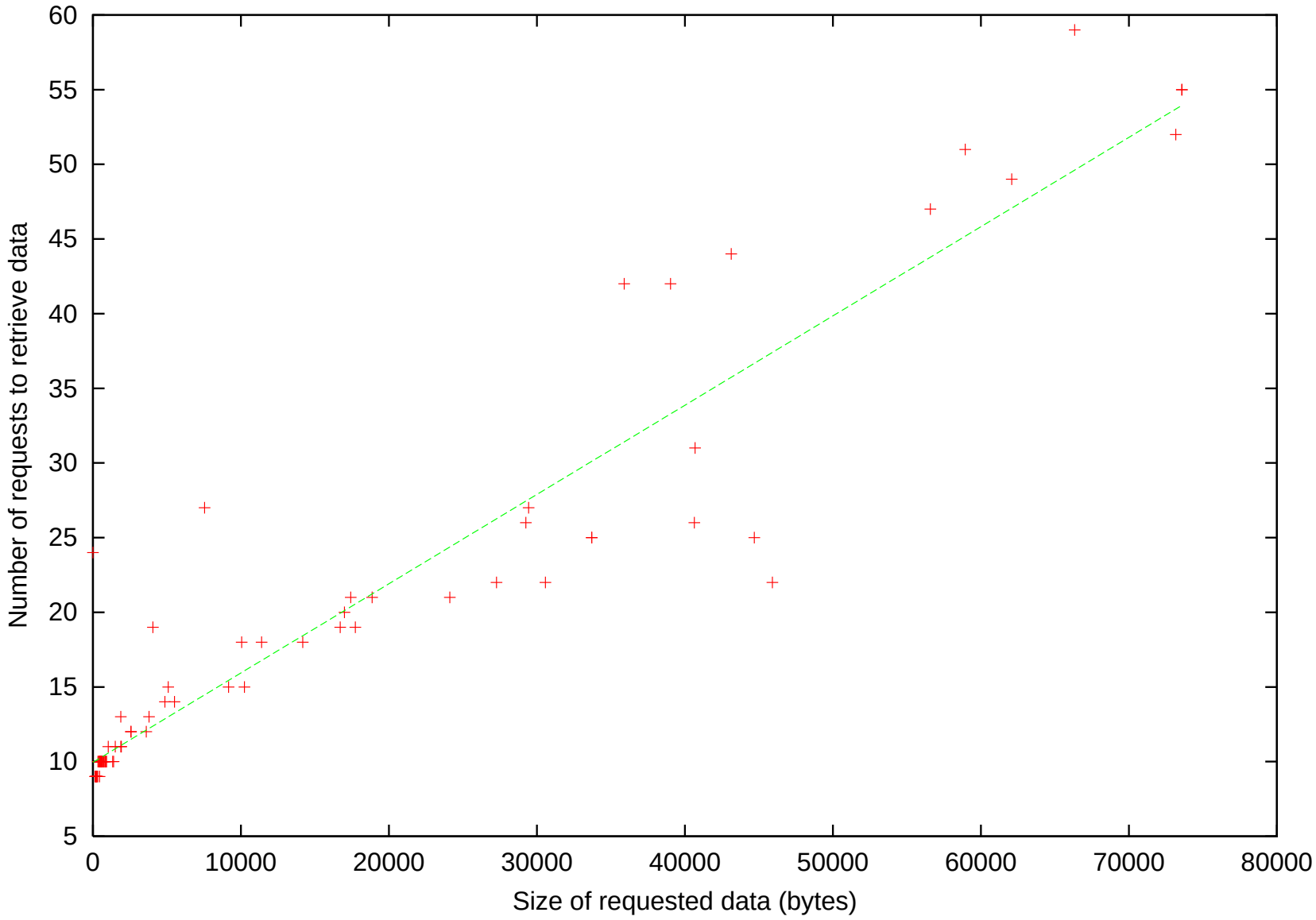
Range-Mapping



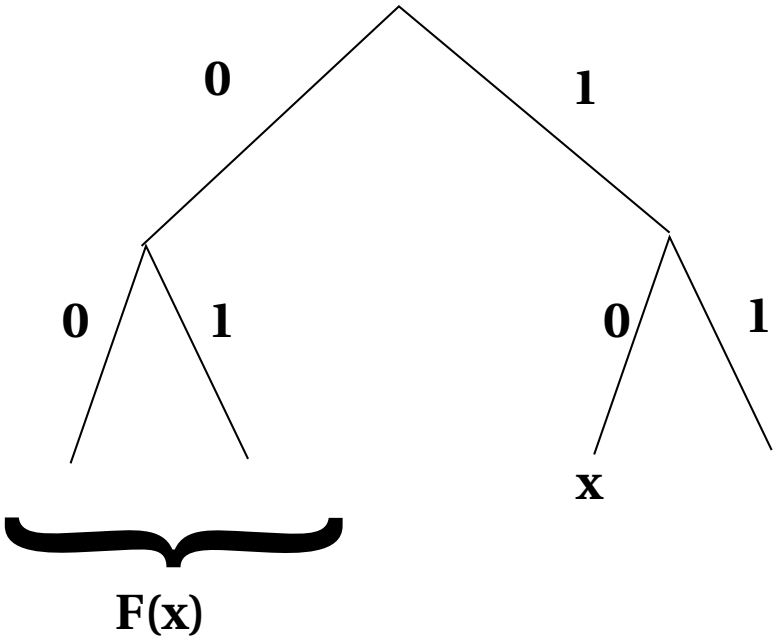
More Visible Links => Greater Upstream Bandwidth

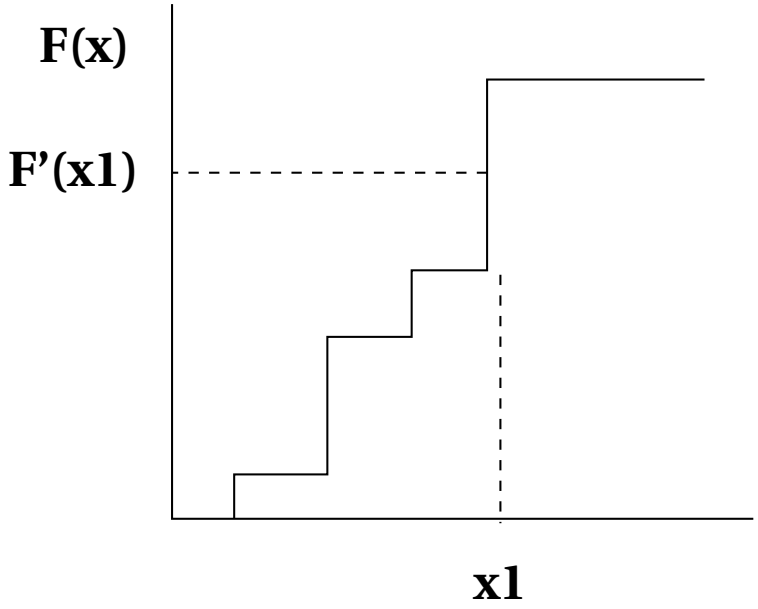


Downstream Performance



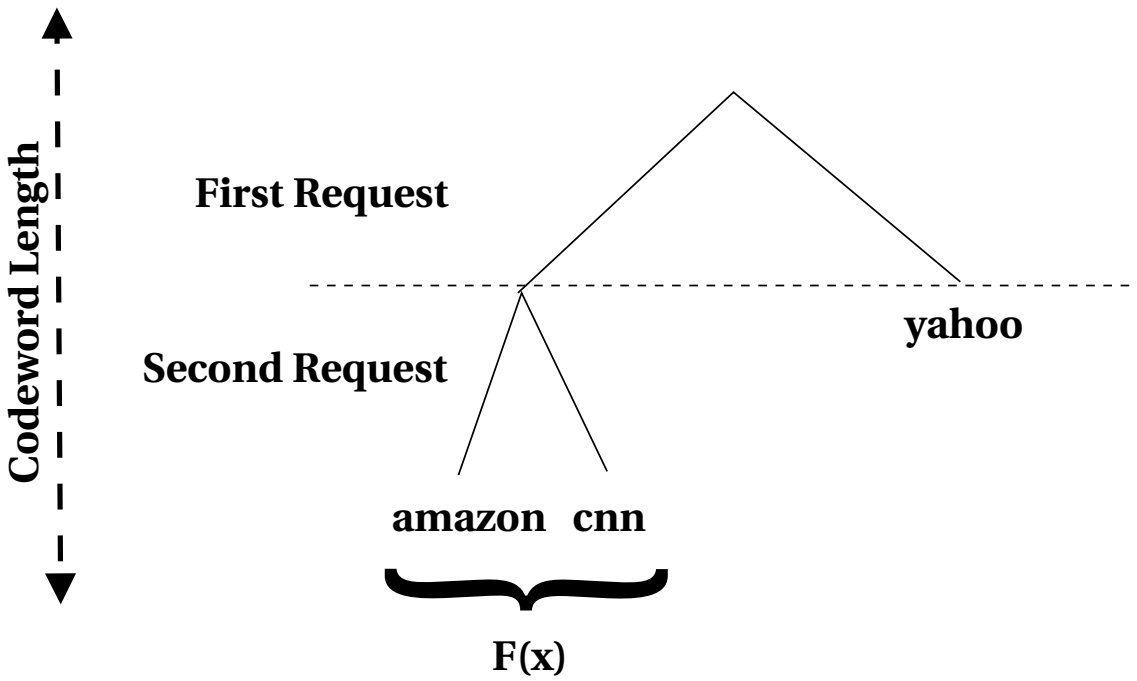
Arithmetic Coding

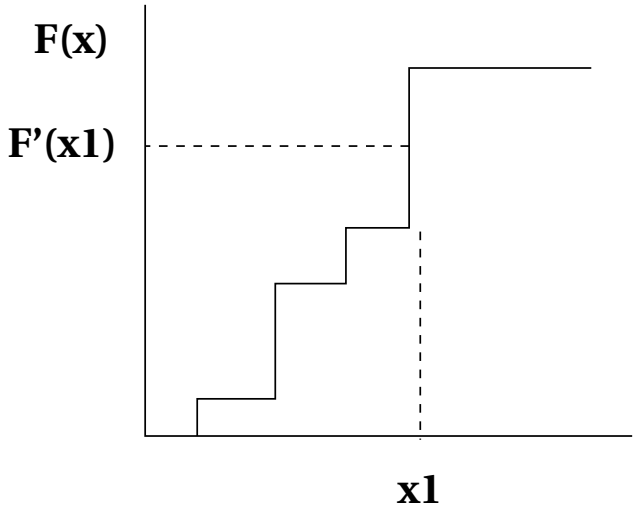




- Encode: Send F'(x)
- Decode: Use decision tree (is F'(x) > P(0)?)
- With a larger range in F(x), F'(x) requires fewer bits to express the range.

Arithmetic Coding





- More popular requests require fewer codewords
- On average, number of requests will be the binary entropy divided by the number of bits per request
- "Skewing" how we divide the distribution: selecting codewords in the distribution.

Steganography

- Two stage process
  - Determine "redundant" bits
  - Select which of these bits to embed the encrypted message (determined by PRG, initialized with seed)
- Set state based on initial seed, embed a secondary seed that selects a PRG that makes minimal modifications
- Defend against statistical attacks that look at distribution of DCT coefficients, etc.

RSA Public Key Cryptosystem

Select p, q prime.

$$\begin{aligned} n &= p \cdot q \\ \phi(n) &= (p-1) \cdot (q-1) \end{aligned}$$

Select an (e, d) such that $ed \equiv 1 \bmod n$. Then:

$$\begin{aligned} C &\equiv M^e \bmod n \\ M &\equiv C^d \bmod n \\ &\equiv M^{ed} \bmod n \\ &\equiv M^{1+k\cdot(p-1)\cdot(q-1)} \bmod n \\ &\equiv M \cdot 1^{k\cdot(q-1)} \bmod p \quad \text{(Euler)} \\ &\equiv M \cdot 1^{k\cdot(p-1)} \bmod q \\ &\equiv M \bmod n \quad \text{(CRT)} \end{aligned}$$