Secure Content Delegation (Blind Cache)
WHAT IT IS
WHERE WE ARE
(SPECS&IMPL)
DISCUSSION
DRAFT-THOMSON-HTTP-SCD

SECURE CONTENT DELEGATION
SPLIT CONTENT AND METADATA
AND HOST CONTENT ANYWHERE

Responses don’t include real content

Content delivered using out of band content encoding

Plus integrity checks

Plus encryption
SLOWER
MAYBE
GO SLOWER

AND MAYBE, LATER, GO FASTER

Big resource, thin pipe, fat pipe
POSSIBLE APPLICATIONS

BIG STUFF

Applicable to distribution of content with large payloads

Video

Large downloads (no need for “official” mirrors)

Maybe down to large images on web pages
SELF DELEGATION
IF YOU WANT SOMETHING DONE RIGHT
DO IT YOURSELF
BUT

...WHY?

C — PROXY — S

NOT SAFE FOR PROXYING

SAFE FOR PROXYING

ENCYPTED

INTEGRITY PROTECTED

ANONYMIZED
... LATER

SHARED CACHING!

C2

PROXY

S
Client makes requests with two indicators:

“\(I\) accept out of band content encoding”

“\(I\) have a proxy handy”

Server decides what to do about that

New signal for out of band: “using a proxy is OK”
OOB ENCODING

Metadata from the origin (primary) server, payload from a cache (secondary resource).

Somewhat equivalent to an HTTP redirect, but

• done on the content coding layer
• preserves the HTTP origin
• Payload allows additional data, such as additional URIs and extensions

Composes with other content codings, such as for encryption.
OMFG
SHARED CACHING?!

All we needed to do was add a new mechanism for content delegation, slap on a whole bunch of crypto, and make a bunch of extra requests, plus a smattering of new signalling

... does it make things faster? Maybe, maybe not

... is it all worthwhile? Quite possibly
WHO NEEDS SERVERS?
THIS FIRST REQUEST IS A REAL DRAG
SPOT THE DIFFERENCE

http://www.flickr.com/photos/24340456@N03/3345977842/

https://en.wikipedia.org/wiki/Orange_(fruit)#/media/File:Orange-Whole-%26-Split.jpg
Lots of request-handling headers, or common values

Accept-Ranges: bytes
Age: 47451
Content-Type: image/jpeg
Strict-Transport-Security: max-age=31536000
Timing-Allow-Origin: *
Via: 1.1 varnish, 1.1 varnish, 1.1 varnish, 1.1 varnish
X-Cache: cp1049 hit(5), cp2005 hit(1), cp4007 hit(2), cp4005 frontend miss(0)
X-Firefox-Spdy: 3.1
X-Timestamp: 1443711458.04701
X-Trans-Id: txe34b67c45534376aeb09-0056fbd60c
access-control-allow-origin: *
access-control-expose-headers: Age, Date, Content-Length, Content-Range, X-

Remainder of metadata is small, and could change infrequently

Last-Modified, Etag, Content-Disposition, and x-object-meta-sha1base36 for these images
SO COMPRESS
A LOT

Without content in every response, h2 server push for large swathes of a site might be possible

Test limits of hpack for very large numbers of resources

  Maybe more practical with a custom format

...work in progress
RESOURCE MAP
AN OOB RESPONSE OPTIMISATION

draft-eriksson-http-resource-map

OOB RESPONSE
“ON-A-STICK”
ORIGIN PUSHES OOB INFO FOR SEVERAL RESOURCES TO CLIENT (RESOURCE MAP)
RUNNING CODE AND TEST BED

SOME TEST RESULTS
Virtual machines running our prototype
• Only link RTTs are emulated

• ON KPI
  • User experience (page load time, networking time)

• On network and topology
  • 2nd’ary servers are closer to client
  • Between client and BC
    • Low latency
    • High bandwidth
  • Between 2nd’ary server(s) and Origin(s) and Client and Origin
    • Low bandwidth
    • High latency
  • 2nd’ary server and client might have same access and network characteristic towards Origin
DIFFERENT DELAYS BETWEEN ORIGIN AND UA

In this setup, the bigger the delay between the origin and the client the higher the gain.

Performance can be improved more if index.html is cached.
RESOURCE SEGMENTATION
RESOURCES SEGMENTATION

Video on Demand

Contains multiple Random Access Points

Integrity Mechanism work on whole resource

If segmenting with independent integrity verification

Random access improved

Segmentation also useful for:

Load Spreading

Simultaneous retrieval from multiple servers

Privacy Improvements