# Globally Distributed Content Delivery

Presented by Will Johnson

# ToC


- **Introduction**
  - **Goals**
  - **Previous Works**
  - **The Big Picture**
- Network Services
- Technical Challenges
- Concluding Remarks

# Goals

- Stop the flash crowding problem!
- Reduce lost revenue
  - Prevent site crashes
  - Reduce overly high latency
- Reduce load at centralized servers
- Serve Content from the Network Edge!

# Previous Works

# Traditional Approaches

- Local Clustering - Several servers on site to handle requests
  - Difficult to scale to the thousands
- Mirroring - Copies of the server, distributed geographically
  - All mirrors must be synchronised
- Multi-homing - Connect the Server to multiple ISPs
  - If one link fails, BGP does not converge quickly to a new one

# Distributed Approaches

- Autonet - Uses logically centralized controls to recompute routes when a route fails
  - What about a more fine-grained approach that remaps in response to health?
- Web Caching - keep data stored in caches on the web
  - Typical use case: users want to read data.  This means faults are more readily tolerated.
  - Cache misses are quite common because of dynamic content
- Depot - system updates and management for large distributed networks
  - Modular packages
  - Internal consistency verification (what about end-to-end?)

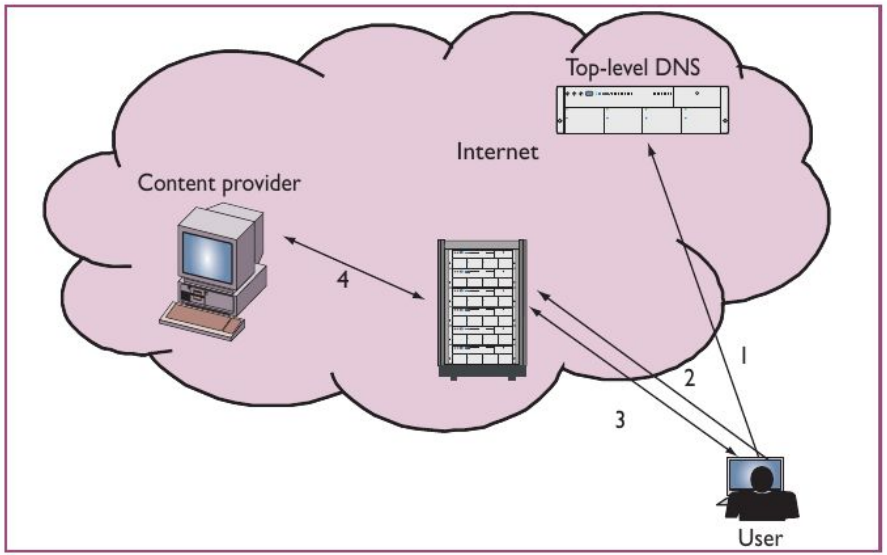
# The Big Picture

# Request Handling

- Akamai handles flash crowds by sending requests to the **nearest**, **available** server that is **likely** to have requested content.
- Nearest - function of network topology and dynamic link characteristics
  - Lower round-trip time
  - Lower packet loss
- Available - function of load and network bandwidth
  - Server with low load, and a datacenter with plenty of bandwidth
- Likely - function of which servers carry the content for each customer in a datacenter
  - Use as few servers as possible

# Mapping

- Mapping - Direction of requests to content servers
    - Based on **Service Requested**, **User Location**, and **Network Status**
    - Uses DNS for load balancing

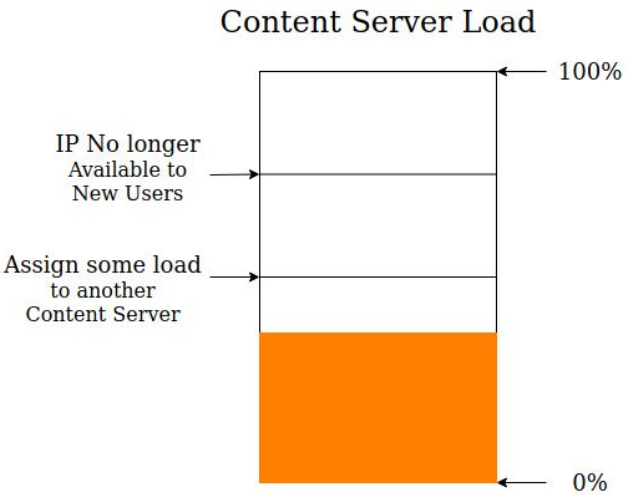


# Mapping Metrics

- Service Requested - Server must be able to service the request
- Server Health - Must be up and running with no errors
- Server Load - Load cannot exceed a certain threshold.
- Network Condition - Data center must have sufficient bandwidth, and packet loss must be minimal
- Client Location - Server must be **Near** to the client
- Content Requested - Server must be likely to have to content requested

# Network Monitoring: Content Server

- Monitoring Application watches loads of each content server
  - Content servers report load
  - Load Data Aggregated and published to DNS



# Network Monitoring: End to End

- Agents simulate end to end behavior
  - Download web objects
  - Measure failure rates
  - Measure download times
- Detect and suspend problematic data centers
- Provide centralized content reporting for data owners

# ToC

- Introduction
- **Network Services**
  - **Static Content**
  - **Dynamic Content**
  - **Streaming Media**
- Technical Challenges
- Concluding Remarks

# Static Content

- Html, pdf, embedded images, executables, etc…
  - Content type helps decide Static or Dynamic
- Static content has labels applied to it
  - Lifetime (from 0 to infinite)
  - Ability to be served over HTTPS
  - Etc…
- Features are decided based on content type, content provider requirements, and other criteria.

# Dynamic Content

- Many modern web pages rely on dynamic content
  - Web caches cannot handle any degree of dynamic content.  Even one dynamic element prevents their use
- Edge Side Includes (ESI) technology allows pages to be broken into static and dynamic content
  - Uses XSLT to provide XML data
  - Similar to Server Side Include languages, but allows for fault tolerance
- Dynamic web pages are re-assembled upon request
- ESI reduces bandwidth requirements for dynamic content by 95 to 99 percent

# Streaming Media

- Supports
  - Windows Media
  - Real
  - Apple Quicktime
- Live streams usually send to an entry point server
  - This should be fault tolerant, and react quickly to an unavailable entry point server
- Entry Point servers then distribute content to Edge Servers
  - Traffic must be routed around congestion
  - Must be resilient against network failures

# ToC

- Introduction
- Network Services
- **Technical Challenges**
  - **System Scalability**
  - **System Reliability**
  - **Software Deployment and Platform Management**
  - **Content Visibility and Control**
- Concluding Remarks

# System Scalability

- For things to run smoothly, the network must support different use cases and geographic locations.
  - Keep bandwidth low, but monitor thousands of servers
  - Monitor and aggregate network health info every few seconds (i.e. keep DNS lookup times down)
  - Gracefully handle outdated programs and information
  - React quickly to new network conditions and loads
  - Handle a wide variety of customer needs
  - Isolate customers
  - Ensure integrity on devices and end-to-end
  - Collect and process logs for billing and analytics

# System Scalability (Cont.)

- To handle this, Akamai developed a distributed, monitoring service
  - Resilient to temporary loss of information
- Customer problems are handled by support teams
  - Diagnostics
  - Billing

# System Reliability

# Fault Tolerance

- System and Network  failures must be identified and fixed quickly
  - DNS automatically detects failures
  - New IP addresses can be handed out
  - Cached Answers can be used
  - All DNS responses have at least 2 IP addresses
  - Backup servers on site can take over for a downed Content Delivery Server
  - The Top Level DNS server can identify DNS servers that have connections

# Fault Tolerance (Cont.)

- Software Flaws must be handled quickly
- New request and response headers emerge all the time
  - Edge servers must interpret these correctly
- Testing new features with the Top Browser and Content Server configs is infeasible
  - Created a test tool that can direct live traffic to test version of software
  - Live traffic allows for faster debugging without affecting the network

# Software Deployment and Platform Management

# Software Updates

- Software must be updated regularly
  - Improved performance
  - Better operational and monitoring capabilities
- Not all servers are available all the time
  - 2 versions of all software will be live at any given time
  - Different components must be able to coexist
- Monitoring software must watch carefully to catch problems

# Device Specifics

- The network runs on the backs of Linux and Windows OSes
  - Monitoring platform must be able to run ubiquitously and collect accurate information
  - Necessary for load balancing
- Diagnostics teams must be familiar with a wide variety of OSes

# Content Visibility and Control

# Content Providers Don’t Own Their Content

- Content Providers should be able to control their data
  - Analytics should be accurate and fine grained
  - Must have the greatest amount of control possible
- This implies that the network must take into account
  - Cache Consistency
  - Lifetime Control
  - Authentication and Authorization
  - Integrity Control
  - Visibility and Billing

# Cache Consistency and Lifetime Control

- Objects that can be cached, should have a Time To Live (TTL) associated with them
  - Some might be cacheable forever
  - Some might be versioned (specified in the URL)
- Cacheable objects should be removable before the TTL has expired
  - Revocation can be done by content providers or network owners
- Non-Cacheable content should travel from the Origin Server to the Edge Server to the User.

# Security

- **Authorized** Content should be protected
  - Edge servers contain authorization features
  - Authorization Tokens can be passed between User and Origin Server
- Content **Integrity** should be enforced, such that corrupted data does not propagate
  - Incomplete responses should not be cached
  - Corrupted data should be detected and re-fetched

# Visibility and Billing

- Content delivery logs should be aggregated to a centralized location
  - Analytics
  - Billing
- This generates a lot of data flow and computation
- Content Delivery Rates and Client Location can be delivered more readily

# ToC

- Introduction
- Network Services
- Technical Challenges
- **Concluding Remarks**
  - **Pros**
  - **Cons**

# Pros

- Works very well for intended use case
  - Huge networks with tremendous amounts of traffic
  - CDNs are still used today!
- Companies that own both the CDN *and* the data suffer no losses
  - They maintain control over their data!

# Cons

- Content providers do not own their content any more
  - Fees to remove, or scrub data from the CDN
- Large startup cost to implement
- Lack of details
  - Distributed Monitoring Application
  - Log Aggregation
  - “95% to 99% reduction in ESI bandwidth usage”
- Paper within a paper?

# Questions