How RSS was won (and where it got us)

Daniel Sandler
and Peter Druschel
Rice University
but first...

A WORD FROM OUR SPONSOR
ATTENTION,
NEWS JUNKIES
Is surfing the web becoming HARD WORK?
TODAY

Thousands of niche websites

Specialized news
Forums and communities
Blogs, blogs, and more blogs

Updating around the clock, around the world!
Somebody could be writing something really interesting
Somebody could be writing something really interesting

RIGHT NOW
Somebody could be writing something really interesting right now

(and you’re missing it)
IS THERE ANY HOPE?
INTRODUCING
INTRODUCING

RSS FEEDS
INTRODUCING RSS FEEDS Web feeds Atom feeds
INTRODUCING

(really simple syndication / rich site summary / RDF site summary)
Katrina Death Toll May Not Hit 10,000

NEW ORLEANS - Alarming predictions of as many as 10,000 dead in New Orleans may have been greatly exaggerated, with authorities saying Friday that the first street-by-street sweep of the swamped city revealed far fewer corpses than feared.

MORE STORIES
- FEMA Dumps Brown As Katrina Relief Chief
- FEMA to Halt Debit Cards, Use Bank Deposit
- Reported Katrina Deaths, State by State
- Small Signs of Normalcy Return in Miss.
- Ophelia, a Hurricane Again, May Hit U.S.

Personalize News Home Page: Add/Remove News Categories | Change Layout | Weather

Top Stories
- Congress to Investigate 9/11 Loan Abuses
Katrina Death Toll May Not Hit 10,000

NEW ORLEANS - Alarming predictions of as many as 10,000 dead in New Orleans may have been greatly exaggerated, with authorities saying Friday that the first street-by-street sweep of the swamped city revealed far fewer corpses than feared.
Subscribe to the Web with RSS today!

Side-effects may include dry mouth, headache, and uncontrolled polling of server resources. Consult your local doctor.
(end of advertising)
Rough agenda

- The RSS pitch
- What could possibly go wrong?
- Our solution: FeedTree
- A live, public p2p network: what we observed
- Takeaways: the good, the bad
RSS, revisited

- XML summary of headlines, links, stories

- Result: dramatically easier to keep up with "micronews" from many sources

- More and more feeds
  - Feedster ca. 2004:
    - 800,000+ feeds
  - Feedster today:
    - 41 million

Feedster ca. 2004:
- 800,000+ feeds
Feedster today:
- 41 million
“Breaking News”

- Each RSS reader must poll every subscribed feed
  - Frequently (for freshness!)
  - Many requests are superfluous
  - Popularity curse (“sticky” traffic)
  → This doesn’t scale! Bandwidth problem for publishers.

- Some attempts to control the problem
  - Scaling back RSS service (e.g. MSDN)
  - Tracking, punishing greedy clients (e.g. Slashdot)
  - “Make it somebody else’s problem” (e.g. FeedBurner)
“Fixing News”

- Perfect fit for peer-to-peer multicast
  - Many participants
  - Continuously running client applications
  - Don’t need extremely low latency
    - Still much better than polling: 10 sec. ≪ 60 min.

- **Goal:** cooperative p2p dissemination of micronews *as it happens*
Propose **FeedTree**, a p2p micronews distribution system built on **Scribe**
- Use Scribe’s multicast service to disseminate RSS news, ASAP, without polling

**Scribe: Batteries included**
- Decentralized membership
- Cheap maintenance of multicast trees
- Efficient multicast messages
Ingredient #1: Pastry, a structured p2p overlay network.
Pastry: Circular ID space

\[(2^{160} - 1)\]
Pastry: Circular base-16 ID space

0xFF…F

0x00…0

0xC0…0

0x80…0

0x40…0
Pastry: Circular base-16 ID space

Node ID = 0x3A9...

0xFF...F
0x00...0
0xC0...0
0x80...0
0x40...0
Pastry: Recursive routing

Node ID=0x3A9…

0x00…0

0xFF…F

0xC0…0

0x80…0

0x40…0
Pastry: Recursive routing
Pastry: Recursive routing

$A = 0x3A9...$

$0xFF...F$

$0xC0...0$

$0x80...0$

$0x00...0$

$0x40...0$
Pastry: Recursive routing

A = 0x3A9...
B = 0x52B...

0x00...0
0xFF...F
0xC0...0
0x40...0
0x80...0
Pastry: Recursive routing

A = 0x3A9...
B = 0x52B...

0x00...0
0xFF...F
0xC0...0
0x80...0
0x40...0
Pastry: Recursive routing

A = 0x3A9...

B = 0x52B...

0x00...0

0xFF...F

0xC0...0

0x80...0

0x40...0
Pastry: Recursive routing

A = 0x3A9...

B = 0x52B...

0xFF...F

0x00...0

0xC0...0

0x40...0

0x80...0
Pastry: Recursive routing

- $0xFF \ldots \text{F}$
- $0xC0 \ldots \text{0}$
- $0x00 \ldots \text{0}$
- $0x40 \ldots \text{0}$
- $0x80 \ldots \text{0}$

A = $0x3A9 \ldots$

B = $0x52B \ldots$
Multicast

- **Ingredient #2**: a way to send a message efficiently to a subset of the whole network

- Specifically, nodes interested in a certain “topic”

- Peer-to-peer *multicast*, using a system called Scribe
- Uses Pastry routing to create message distribution trees
- Root = node whose ID is nearest to $\text{HASH}(\text{topic})$
- Tree = $[\text{UNION(all routes to group root)}]^R$
Many Trees Make Light Work

- One ring, many trees
- Low maintenance cost
  - Group trees embedded in existing overlay graph
  - Cheap joins
  - Keep-alive messages subsumed by traffic
  - Each node is interior in 1/16 of its trees
FeedTree Architecture

- Each RSS feed gets a Scribe group
- Publishers
  - Multicast micronews immediately
- FeedTree aware clients
  - Subscribe to multicast groups for each feed
  - Rely on multicast for all news content delivery
- Conventional RSS feed
  - URL provides Scribe topic
  - Recovery of recent lost items
Incremental Adoption

- Compatibility with existing feeds and clients is **essential**
- Early adopters use existing RSS clients with **ft-proxy**, a local HTTP proxy that connects to FeedTree
- Proxies look for relevant Scribe groups to join
  - If no group exists, **poll** the feed as usual, and start a group, multicasting each new item. Share your work with others!

[Image of a RSS feed with a link to slashdot.rss]

**http://slashdot.org/ slashdot.rss**

**POLL**

(HTTPS)
Incremental Adoption

- Compatibility with existing feeds and clients is essential
- Early adopters use existing RSS clients with ft-proxy, a local HTTP proxy that connects to FeedTree
- Proxies look for relevant Scribe groups to join
  - If no group exists, poll the feed as usual, and start a group, multicasting each new item. Share your work with others!
Incremental Adoption

- Compatibility with existing feeds and clients is **essential**
- Early adopters use existing RSS clients with **ft-proxy**, a local HTTP proxy that connects to FeedTree
- Proxies look for relevant Scribe groups to join
  - If no group exists, **poll** the feed as usual, and start a group, multicasting each new item. Share your work with others!
Multiple **ft-proxy** instances cooperatively poll the same legacy feed for better event resolution
Multiple **ft-proxy** instances cooperatively poll the same legacy feed for better event resolution

http://slashdot.org/slashdot.rss
Multiple **ft-proxy** instances cooperatively poll the same legacy feed for better event resolution

http://slashdot.org/
slashdot.rss

N=1
T=30min
Multiple **ft-proxy** instances cooperatively poll the same legacy feed for better event resolution
Multiple **ft-proxy** instances cooperatively poll the same legacy feed for better event resolution

http://slashdot.org/slashdot.rss

N=2
T≈15min
Multiple **ft-proxy** instances cooperatively poll the same legacy feed for better event resolution.

http://slashdot.org/
slashdot.rss

N=2
T≈15min
Multiple \textit{ft-proxy} instances cooperatively poll the same legacy feed for better event resolution.
The FeedTree network

FeedTree subscriber A

FeedTree publisher B

Legacy feed publisher E

FeedTree subscriber C

FeedTree subscriber D

HTTP request

signed multicast

multicast
Everyone Loves Screenshots

ft-proxy
web interface

<table>
<thead>
<tr>
<th>Feed</th>
<th>ID</th>
<th>Cached entries</th>
<th>Mode</th>
<th>Timeline</th>
<th>Events seen</th>
<th>Entries seen</th>
<th>Sources</th>
<th>Next period</th>
</tr>
</thead>
<tbody>
<tr>
<td>BBC News</td>
<td>0x334466..&gt;</td>
<td>30</td>
<td>polling</td>
<td>30m</td>
<td>6</td>
<td>4.33/hr</td>
<td>33</td>
<td>2</td>
</tr>
<tr>
<td>Boing Boing</td>
<td>0x780102..&gt;</td>
<td>30</td>
<td>polling</td>
<td>30m</td>
<td>7</td>
<td>5.05/hr</td>
<td>30</td>
<td>0</td>
</tr>
<tr>
<td>CNET News.com</td>
<td>0x363766..&gt;</td>
<td>6</td>
<td>polling</td>
<td>30m</td>
<td>7</td>
<td>5.06/hr</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>CNN.com</td>
<td>0x840b71..&gt;</td>
<td>8</td>
<td>polling</td>
<td>30m</td>
<td>5</td>
<td>3.65/hr</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>digg</td>
<td>0x8d8904..&gt;</td>
<td>30</td>
<td>polling</td>
<td>30m</td>
<td>25</td>
<td>18.11/hr</td>
<td>66</td>
<td>1</td>
</tr>
<tr>
<td>digg / dig</td>
<td>0x3409ac..&gt;</td>
<td>30</td>
<td>polling</td>
<td>30m</td>
<td>22</td>
<td>15.94/hr</td>
<td>92</td>
<td>1</td>
</tr>
<tr>
<td>Digg Blog</td>
<td>0x3b2244..&gt;</td>
<td>6</td>
<td>polling</td>
<td>30m</td>
<td>7</td>
<td>5.08/hr</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>dsandler.org</td>
<td>0x3135x1..&gt;</td>
<td>25</td>
<td>listening</td>
<td>30m</td>
<td>14</td>
<td>5.27/hr</td>
<td>40</td>
<td>0</td>
</tr>
</tbody>
</table>
Everyone Loves Screenshots

---

**ft-proxy**

Web interface:

![Web interface screenshot](image)

**Statistical Information**

- Feeds: 30 (polling 29)
- Entries: 585
- Memory: 34.6M (6% free)
- Uptime: 2h49m43s

**Statistics**

![Statistics](image)

**Feed Information**

(name, URL, homepage)

<table>
<thead>
<tr>
<th>Feed</th>
<th>ID</th>
<th>Cached entries</th>
<th>Mode</th>
<th>Timeline</th>
<th>Events seen</th>
<th>Entries seen</th>
<th>Sources</th>
<th>Next period</th>
</tr>
</thead>
<tbody>
<tr>
<td>BBC News</td>
<td>&lt;0x304466&gt;</td>
<td>30</td>
<td>polling</td>
<td>30m</td>
<td>6</td>
<td>23.82/hr</td>
<td>2</td>
<td>2/20/06 1:09 AM</td>
</tr>
<tr>
<td>Boing Boing</td>
<td>&lt;0x7b8102&gt;</td>
<td>30</td>
<td>polling</td>
<td>30m</td>
<td>7</td>
<td>21.64/hr</td>
<td>0</td>
<td>2/20/06 1:20 AM</td>
</tr>
<tr>
<td>CNET News.com</td>
<td>&lt;0x363766&gt;</td>
<td>6</td>
<td>polling</td>
<td>30m</td>
<td>7</td>
<td>4.34/hr</td>
<td>0</td>
<td>2/20/06 1:22 AM</td>
</tr>
<tr>
<td>CNN.com</td>
<td>&lt;0x8a10d7&gt;</td>
<td>8</td>
<td>polling</td>
<td>30m</td>
<td>5</td>
<td>5.84/hr</td>
<td>0</td>
<td>2/20/06 1:26 AM</td>
</tr>
<tr>
<td>digg</td>
<td>&lt;0x8d984&gt;</td>
<td>30</td>
<td>polling</td>
<td>30m</td>
<td>25</td>
<td>47.8/hr</td>
<td>1</td>
<td>2/20/06 1:25 AM</td>
</tr>
<tr>
<td>digg / dig</td>
<td>&lt;0x8d984&gt;</td>
<td>30</td>
<td>polling</td>
<td>30m</td>
<td>22</td>
<td>66.67/hr</td>
<td>1</td>
<td>2/20/06 1:22 AM</td>
</tr>
<tr>
<td>Digg Blog</td>
<td>&lt;0x10d984&gt;</td>
<td>30</td>
<td>polling</td>
<td>30m</td>
<td>14</td>
<td>15.05/hr</td>
<td>0</td>
<td>2/20/06 1:22 AM</td>
</tr>
<tr>
<td>listening</td>
<td>&lt;0x10d984&gt;</td>
<td>30</td>
<td>polling</td>
<td>30m</td>
<td>7</td>
<td>5.8/hr</td>
<td>3</td>
<td>2/20/06 1:03 AM</td>
</tr>
</tbody>
</table>

**Feed Information Box**

Feed info
(name, URL, homepage)
Everyone Loves Screenshots

ft-proxy
web interface

Scribe group

Feed info
(name, URL, homepage)
Everyone Loves Screenshots

ft-proxy
web interface

FEEDS

Feed info (name, URL, homepage)

Cached items

Scribe group
Everyone Loves Screenshots

**ft-proxy**

**web interface**

---

### Feed info
(name, URL, homepage)

### Feed

<table>
<thead>
<tr>
<th>Feed</th>
<th>ID</th>
<th>Cached entries</th>
<th>Mode</th>
<th>Timeline</th>
<th>Events seen</th>
<th>Entries seen</th>
<th>Sources</th>
<th>Next period</th>
</tr>
</thead>
<tbody>
<tr>
<td>BBC News</td>
<td>0x34466</td>
<td>30</td>
<td>polling</td>
<td></td>
<td>6</td>
<td>33</td>
<td>2</td>
<td>2/20/06 1:09 AM</td>
</tr>
<tr>
<td>Boing Boing</td>
<td>0x7B052</td>
<td>30</td>
<td>polling</td>
<td></td>
<td>7</td>
<td>30</td>
<td>0</td>
<td>2/20/06 1:20 AM</td>
</tr>
<tr>
<td>CNET News.com</td>
<td>0x36376</td>
<td>6</td>
<td>polling</td>
<td></td>
<td>7</td>
<td>6</td>
<td>0</td>
<td>2/20/06 1:22 AM</td>
</tr>
<tr>
<td>CNN.com</td>
<td></td>
<td></td>
<td>polling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>digg</td>
<td>0x88984</td>
<td>30</td>
<td>polling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Digg Blog</td>
<td></td>
<td></td>
<td>polling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>dsandler.org</td>
<td></td>
<td></td>
<td>polling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Scribe group

### Volunteer mode
Are we polling this feed?

### Cached items
Everyone Loves Screenshots

**ft-proxy**
web interface

**Event timeline**
FeedTree messages seen during the last period

- **Scribe group**
- **Cached items**
- **Feed info**
  (name, URL, homepage)
- **Volunteer mode**
  Are we polling this feed?
Everyone Loves Screenshots

ft-proxy
web interface

Event timeline
FeedTree messages seen during the last period
30m

Volunteer mode
Are we polling this feed?

Feed info
(name, URL, homepage)

Scribe group

Cached items

Event rate
Publisher Adoption of FeedTree

- Desirable for content authentication
- Publishers must join the network
  - **Incremental:** Add a server-side proxy
    - polls existing RSS (very frequently)
    - multicasts cryptographically signed updates
      - `ft-publisher` software
  - **Eventually:** Integrate FeedTree with RSS-aware publishing tools
    - instant multicasts; no polling
Motivations for Adoption

- Cheaper, richer offerings for publishers
  - Dramatically lower bandwidth demands
  - Offer richer or differentiated content
    - As compared with conventional RSS

- Better RSS service to end users
  - Immediate delivery
  - Richer content if offered by publishers
USERS.

(Where do we find them?)
Faster Feeds Using FeedTree Peer-To-Peer

Posted by ScuttleMonkey on Mon 20 Feb 01:49PM from the instant-gratification-generation dept.

dsandler writes

"Researchers at Rice University have just released version 0.7 of FeedTree, a peer-to-peer system for distributing Web feeds faster. Instead of polling feeds independently, FeedTree users cooperate to share news updates using multicast in Pastry, a scalable p2p overlay network. FeedTree reduces the update delay for existing RSS and Atom feeds to a few minutes without putting extra stress on the webserver (anyone who's ever been temporarily banned by Slashdot's RSS feed knows this is a real concern). Feed publishers can also choose to push digitally signed updates for immediate, tamper-proof delivery to subscribers. The client software (download) runs on Linux, OS X, and Windows, and works with any desktop feed reader."
The day before the Slashdot story:

- 3 distinct (non-Rice) IP addresses participating in the FeedTree network
Adoption

- The day before the Slashdot story:
  - 3 distinct (non-Rice) IP addresses participating in the FeedTree network

- The next day:
  - 54 distinct IPs
FeedTree adoption: Web hits and network size

- **Network size**: Distinct IPs
- **HTTP**: homepage HTML
- **HTTP**: blog HTML

- **Feb 20**: v0.7.0 (Slashdot article)
- **Mar 2**: v0.7.1
- **Mar 14**: v0.7.2
A real-world Scribe tree for a feed
Subscription popularity

Feed (ranked by number of subscribers)

Subscribers

Zipf curve ($\alpha = -0.36$)
Subscription popularity

Number of subscribers vs. Feed (ranked by number of subscribers).

Zipf curve ($\alpha = -0.36$)

Subscribers: slashdot.org RSS
Subscription popularity

Number of subscribers vs. feed (ranked by number of subscribers). The graph shows a Zipf curve ($\alpha = -0.36$) for subscribers of slashdot.org RSS feeds.
Network traffic

Feed data transferred (MB/day)

Network size: distinct IPs

Nodes 1 & 2
Nodes 3 & 4
Node 5
Bandwidth of polling nodes

Feed data transferred (MB/day)

Feb 25  Mar 04  Mar 11  Mar 18  Mar 25

Node 3

Node 4

43 MB

Node 3

Node 4
Server bandwidth

Feed data transferred (MB/day)

Slashdot RSS traffic
FeedTree traffic (publisher node)
Where did it get us?

- **RSS**
  - Users love it (quite popular!)
  - Polling doesn’t scale
  - Bandwidth = death of a thousand cuts for publishers

- **p2p content distribution**
  - Benefits for publishers and users
  - Long-term scalability
  - A component of the “right” solution

- We saw an opportunity...
feedtree: collaborative micronews delivery.

FeedTree brings news feed updates to users, instead of the other way around. FeedTree users work together to share the latest Web feed updates. News arrives immediately, instead of on a polling schedule, making RSS as instantaneous as e-mail. Learn more.

FeedTree helps publishers control bandwidth. Peers in the FeedTree network cooperate to share the burden of delivering news, allowing publishers to offer larger news feeds with more frequent updates. Publishers who choose to push content directly to the FeedTree network ensure immediate and authentic delivery to their readers. Learn more.

How does this work? The short answer is that new RSS items are distributed using peer-to-peer multicast. For the long answer, including detailed notes on security, incremental deployment, and research topics, consult the introduction to FeedTree (or print out the poster).

get started.

Readers: read more about FeedTree for users, and download software to connect your existing news reading software to the FeedTree network. (screenshots)
Conclusions (1)

- **The good: It works!**
  - A real, live FreePastry application
  - No PlanetLab crutches necessary
  - Users receive equal or better service
  - Compatible with all legacy feeds, all existing feed reading apps
  - The system pretty much runs itself, even today
    (...another FreePastry first?)
Conclusions (2)

- The bad: Nobody uses it!
  - FreePastry isn’t ready for Joe MySpace
    - Behind a NAT you don’t control? Out of luck.
  - Chicken-and-egg incentives
    - “Faster feeds” not enough to sell users on it
    - “Lower bandwidth” not enticing to publishers when there are VC-soaked firms (FeedBurner) ready to absorb the bandwidth

- The “RSS bandwidth problem” is still out there...
fin