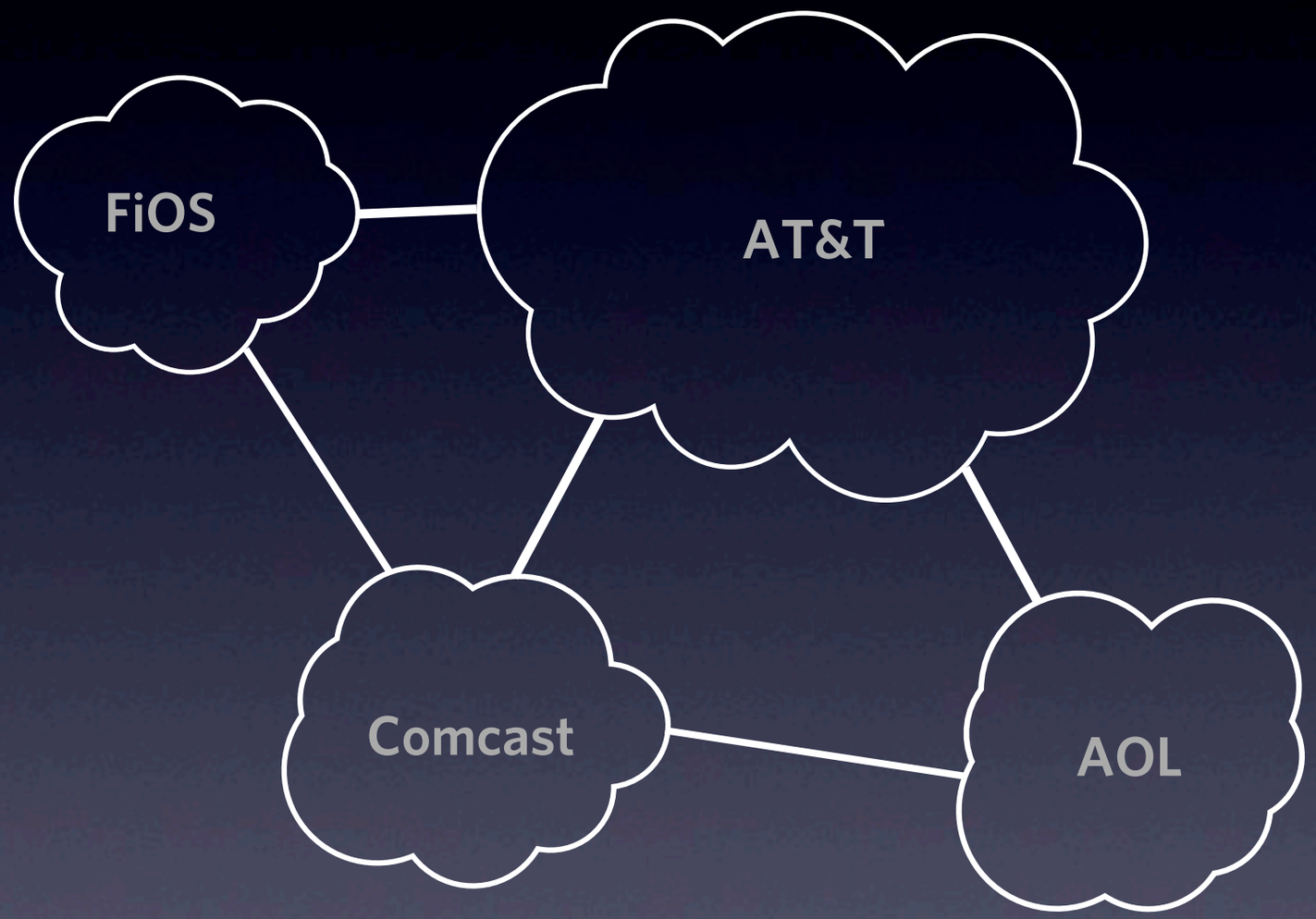
WebCloud: Recruiting social network users to assist in content distribution

Fangfei Zhou[†] Liang Zhang[†] Eric Franco[†] Alan Mislove[†]
Richard Revis[‡] Ravi Sundaram[†]

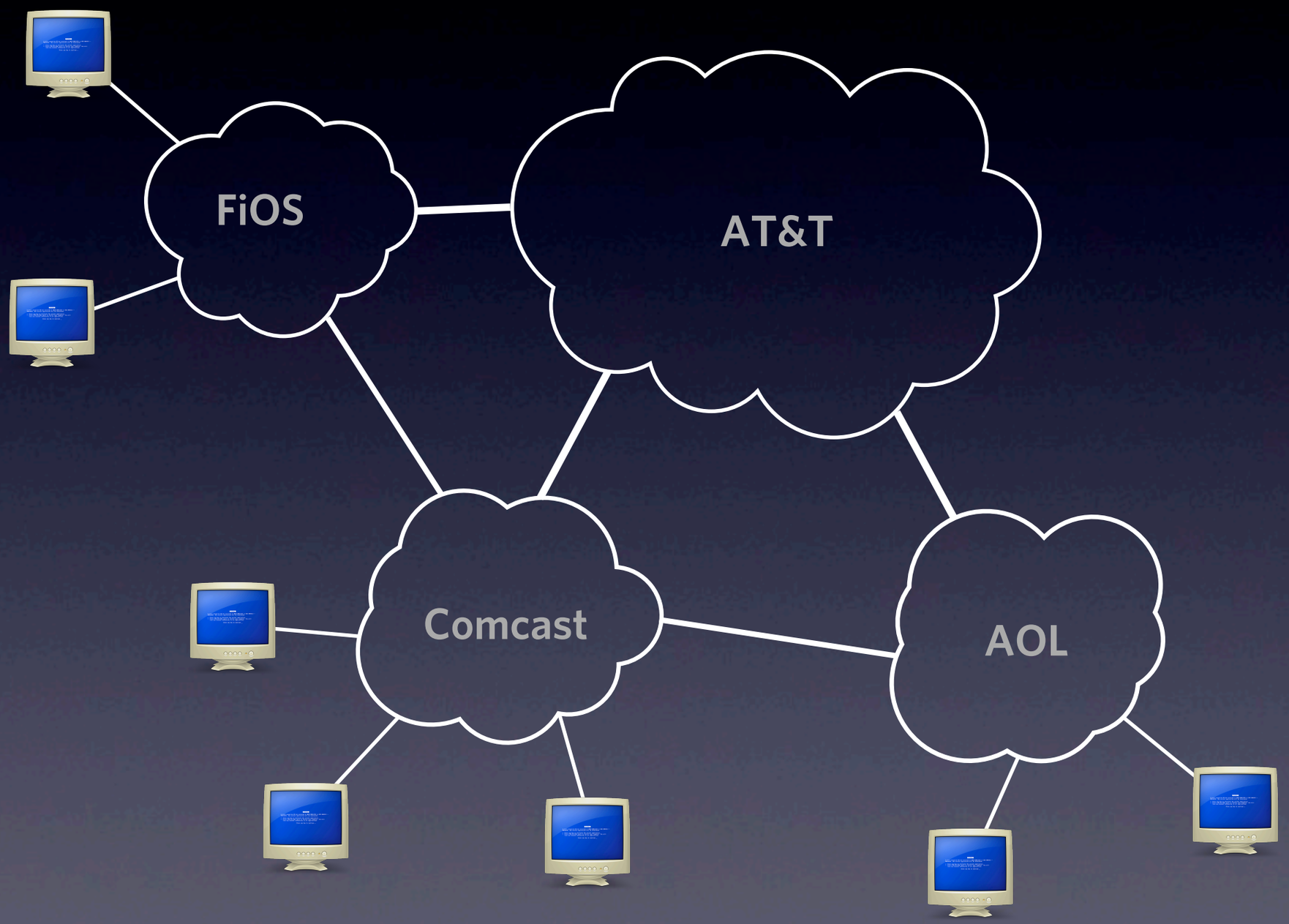
[†]*Northeastern University* [‡]*Jandrell, Pearson & Revis Ltd.*

August 23, 2012, NCA'12

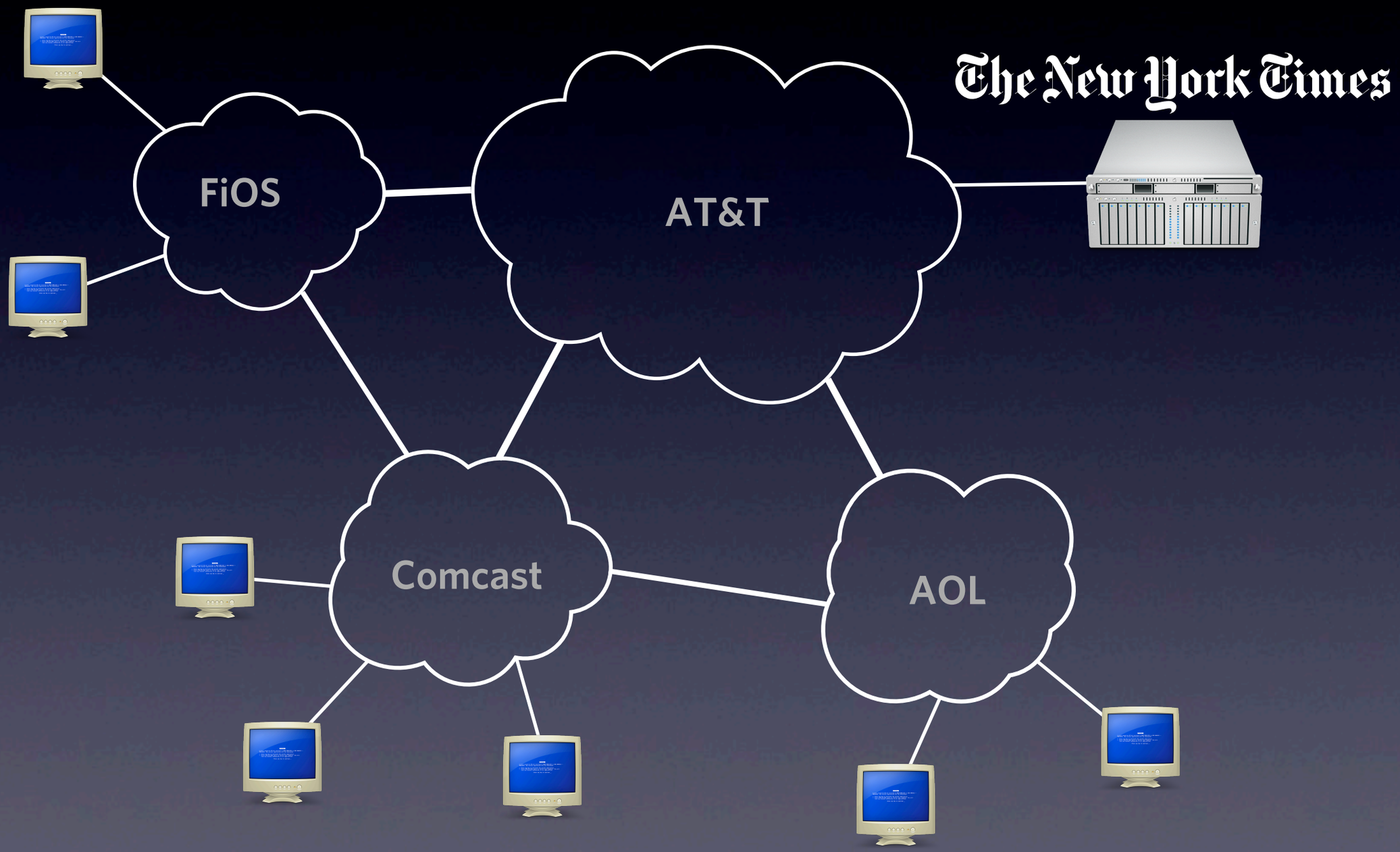
“Classic” web content distribution



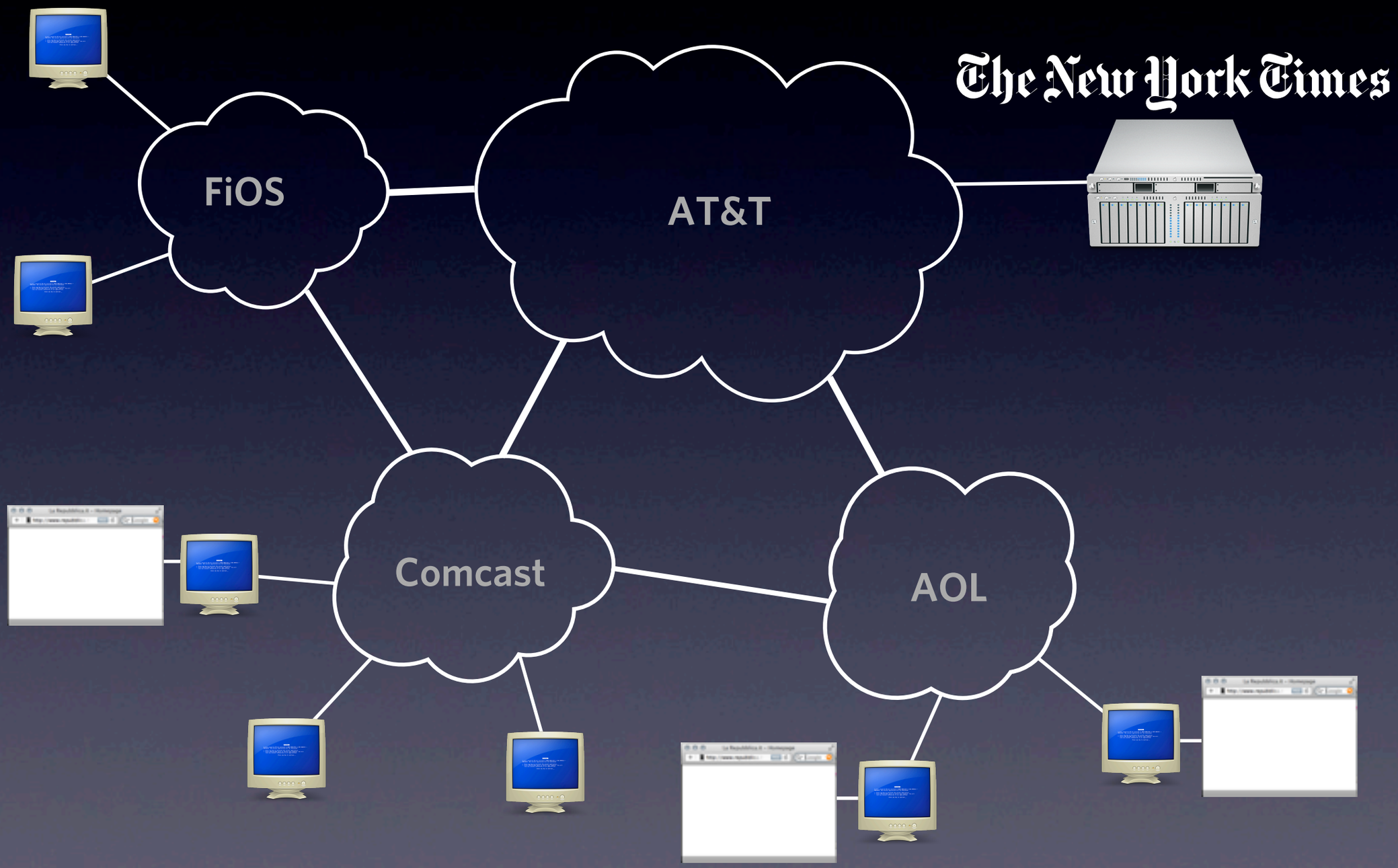
“Classic” web content distribution



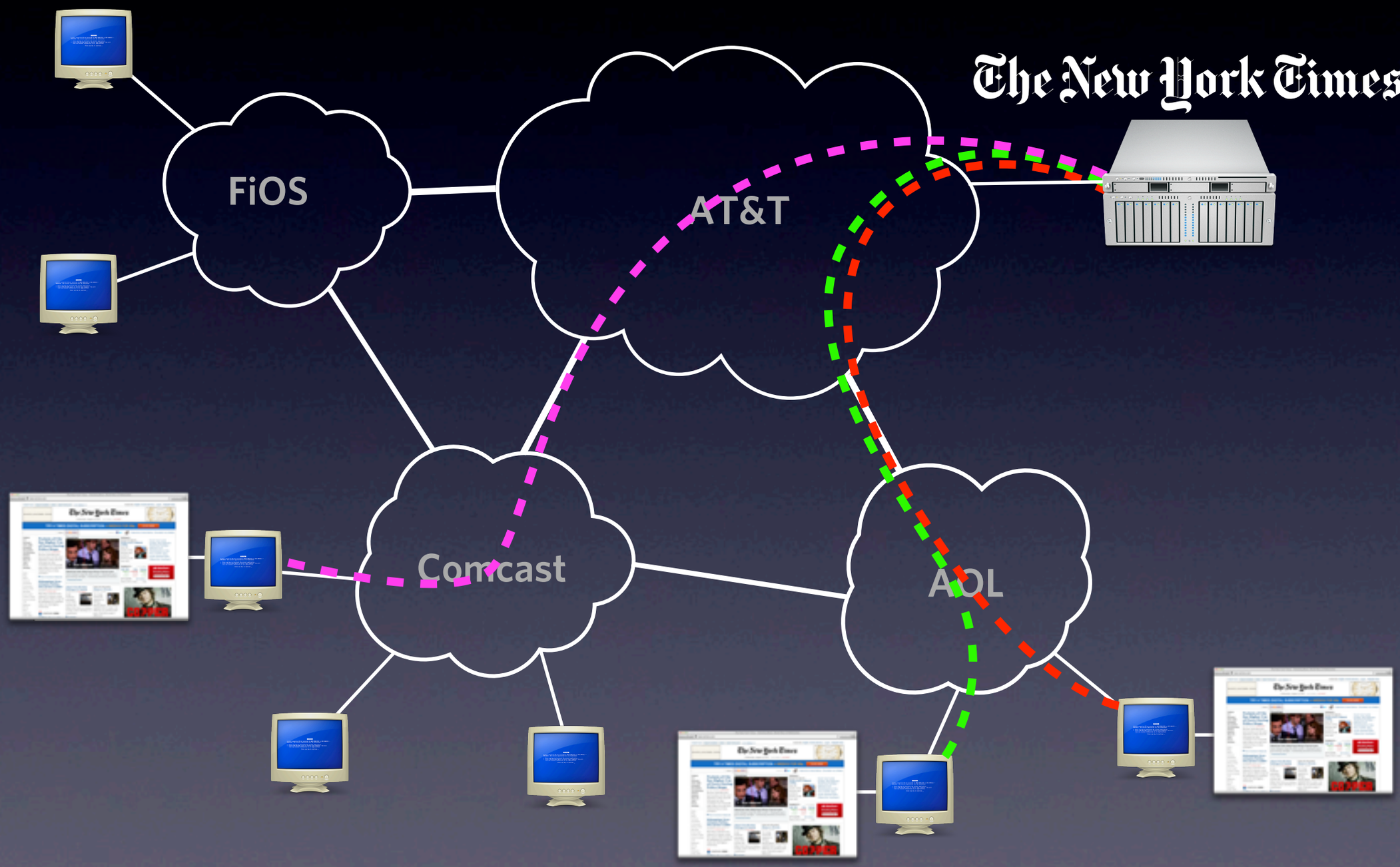
“Classic” web content distribution



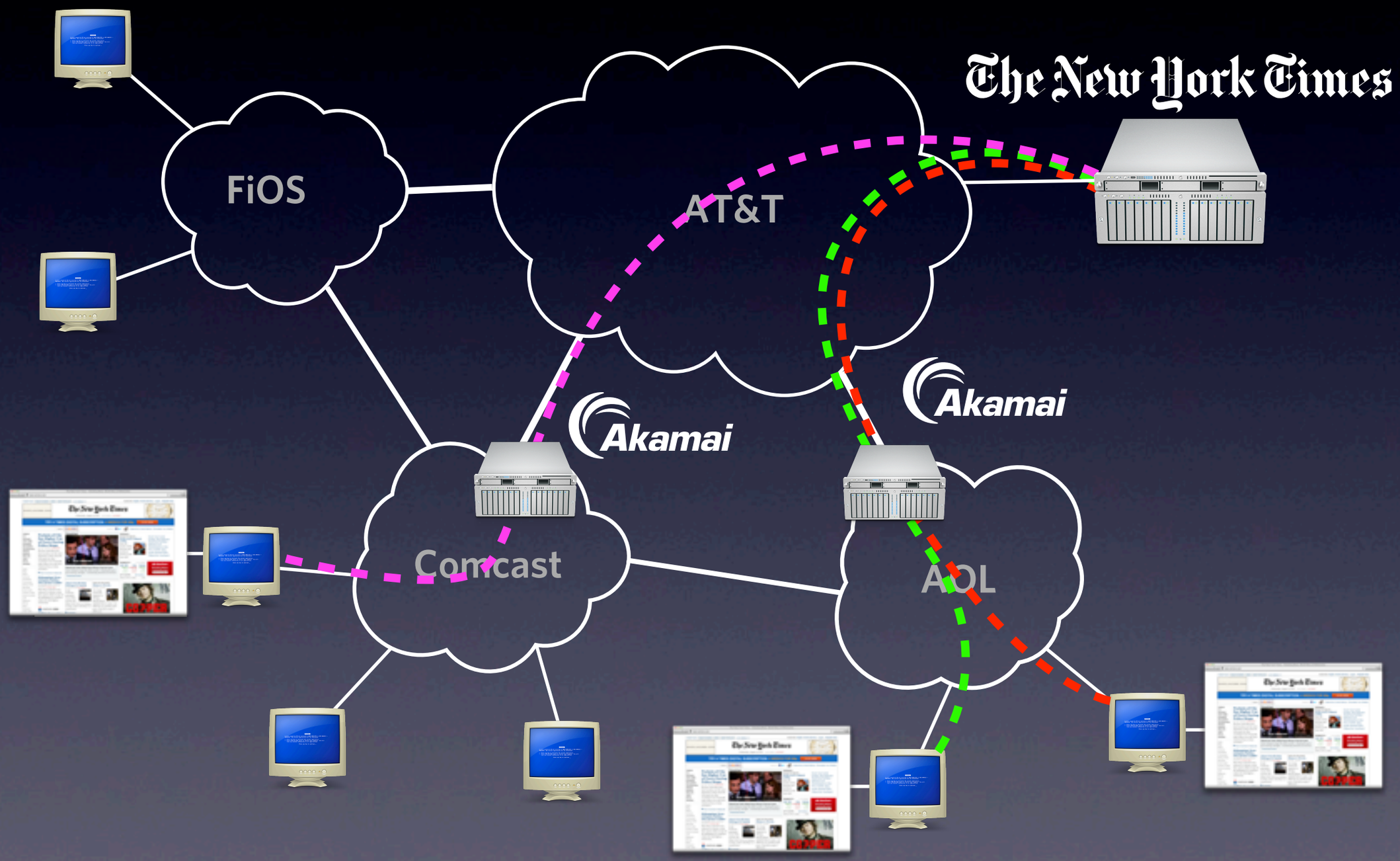
“Classic” web content distribution



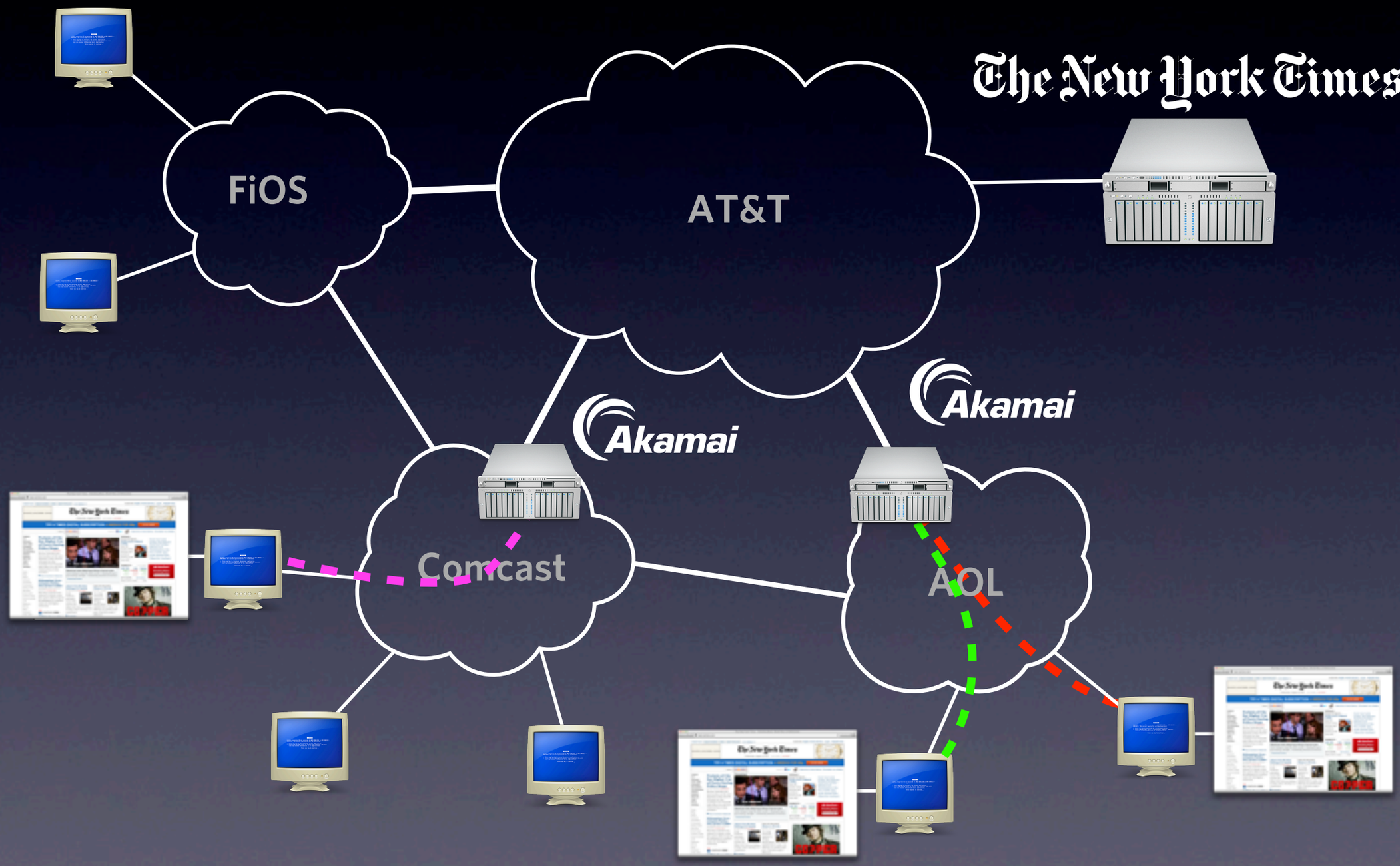
“Classic” web content distribution



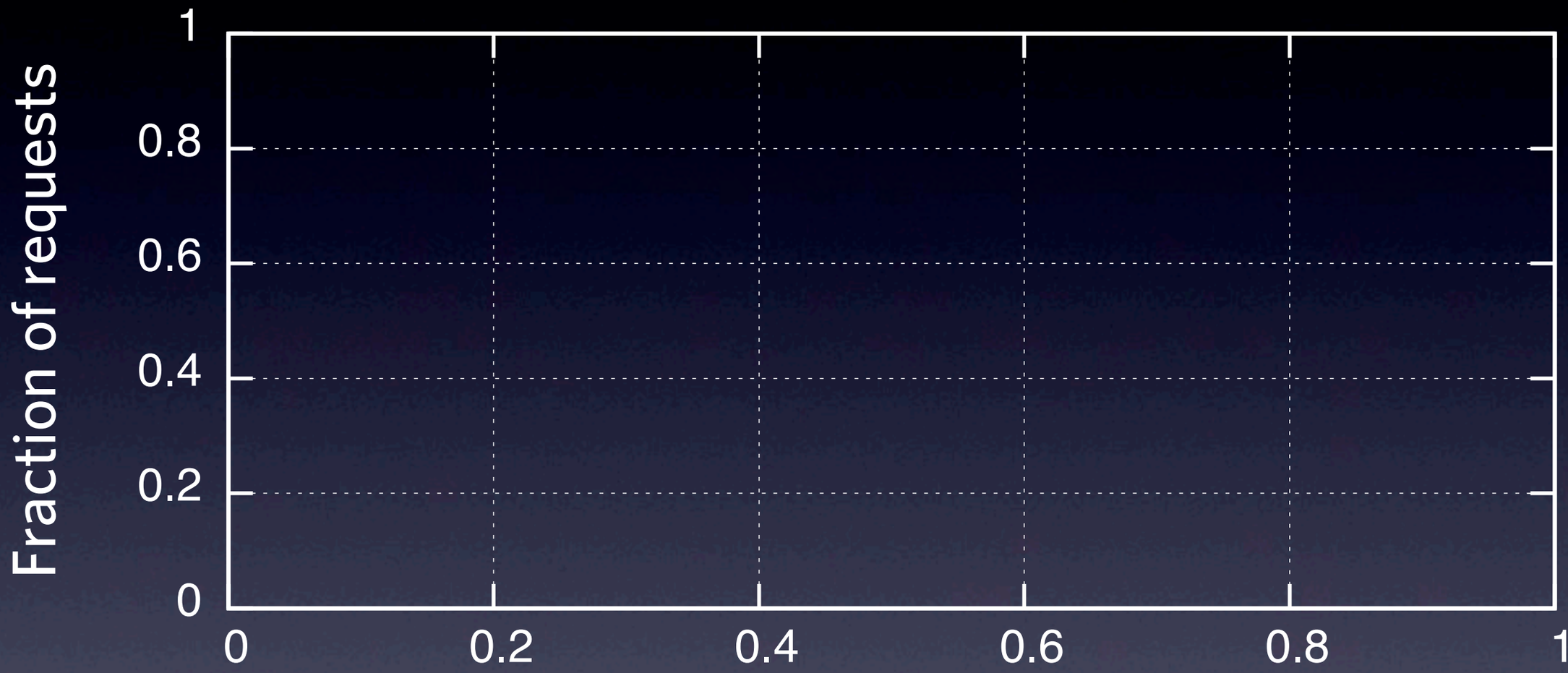
“Classic” web content distribution



“Classic” web content distribution

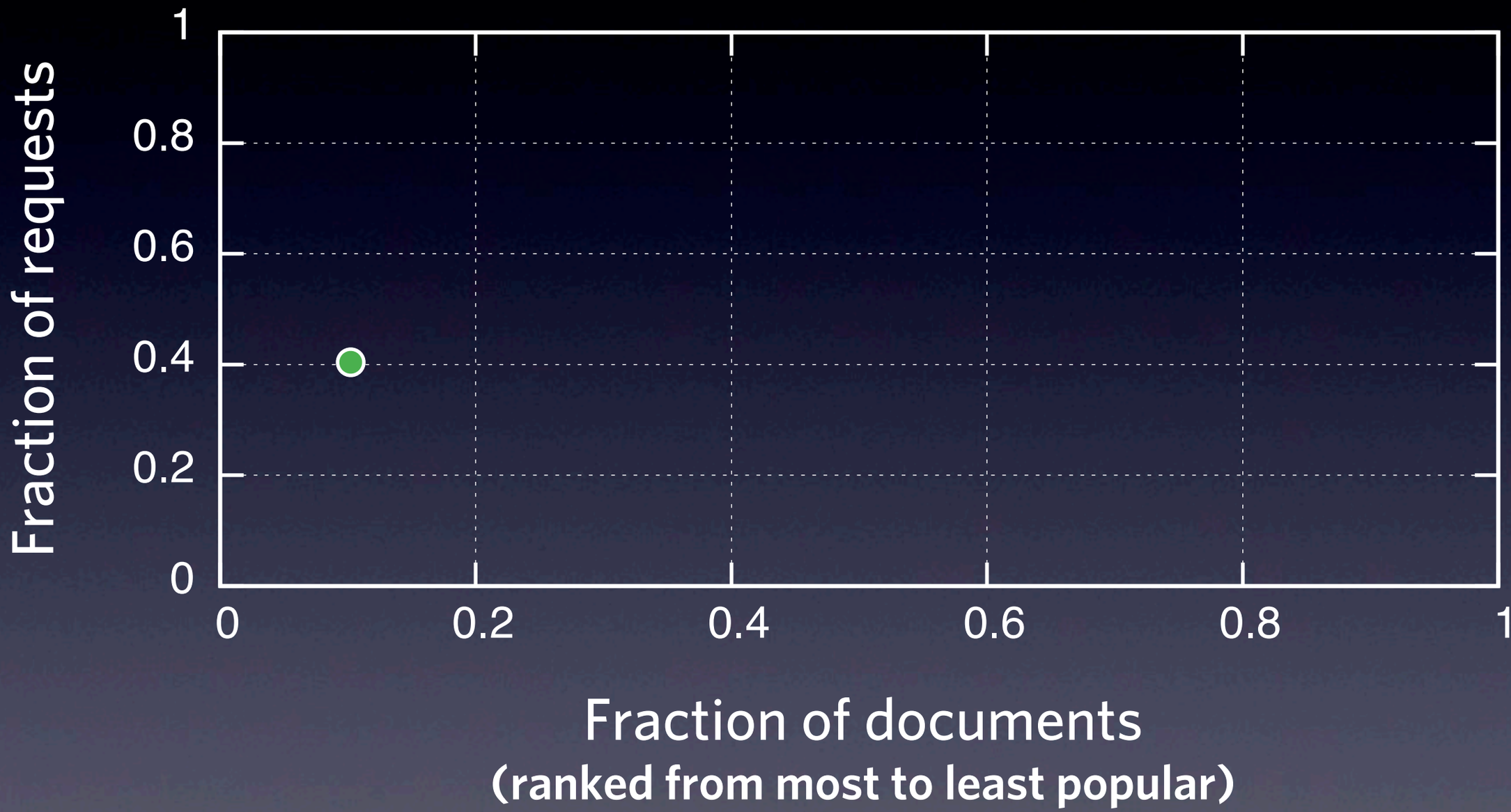


Classic web and OSN content popularity



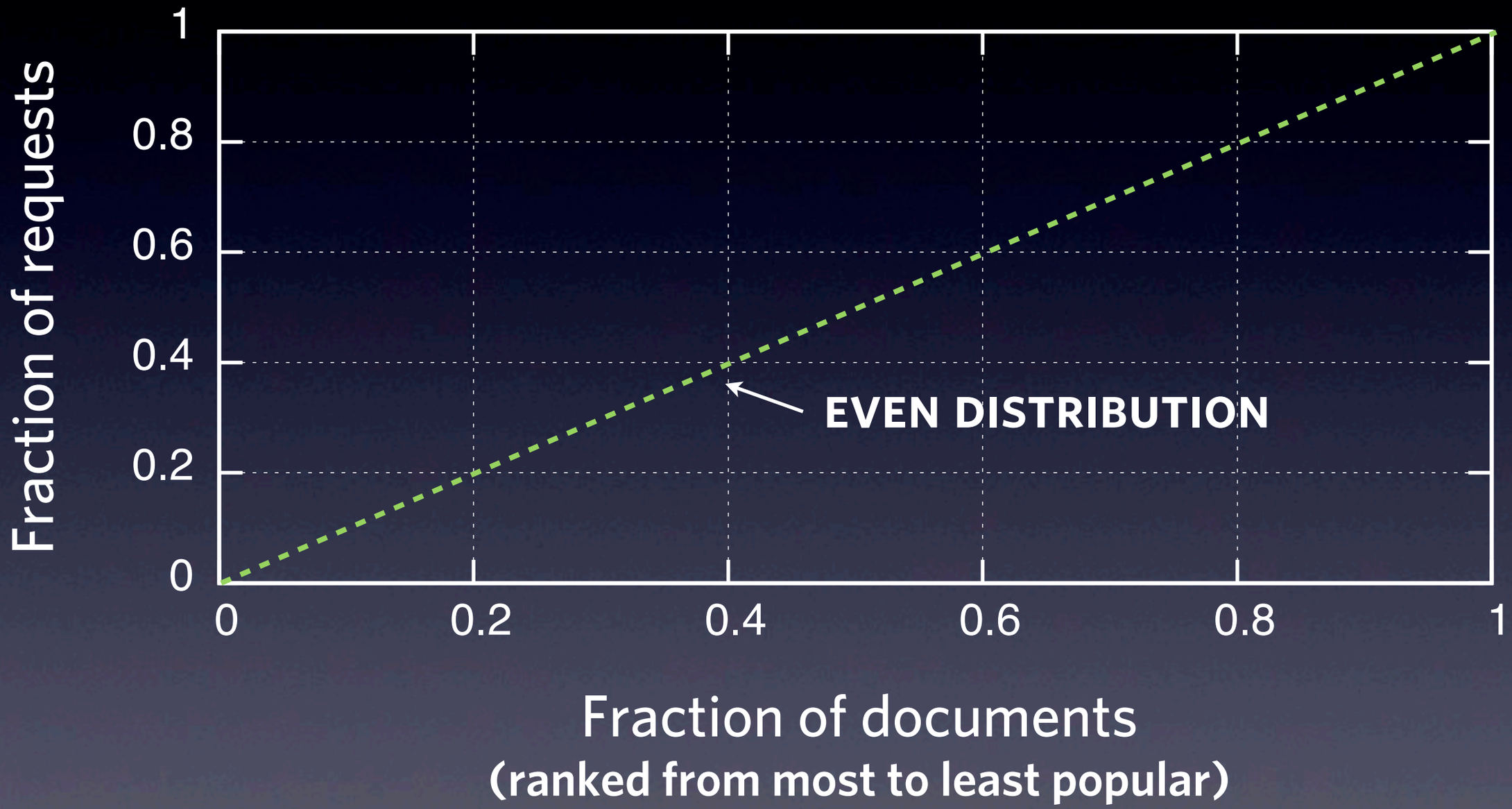
[1] Breslau et al., INFOCOM, 1999, [2] Mislove et al., WSDM, 2010

Classic web and OSN content popularity



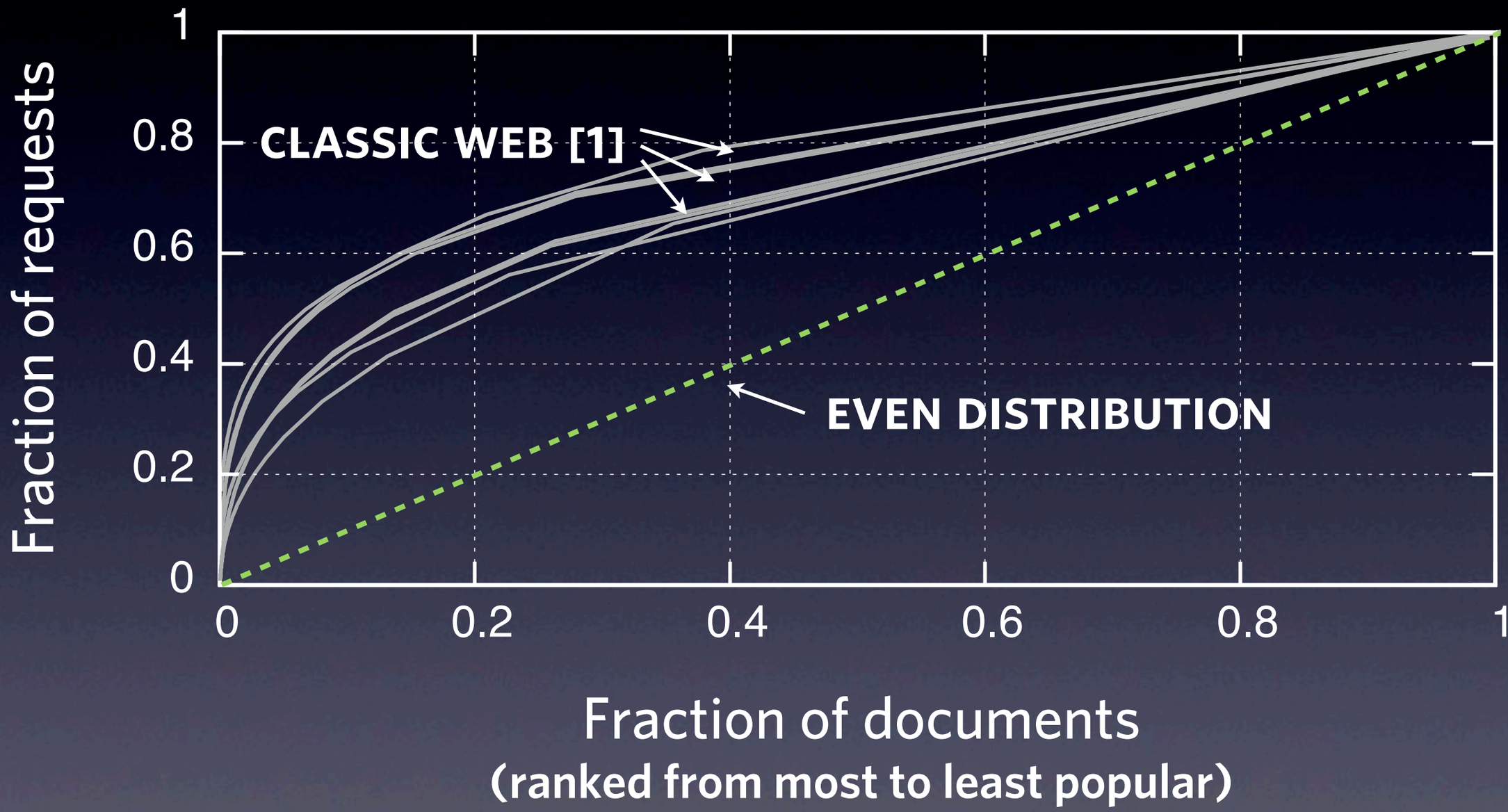
[1] Breslau et al., INFOCOM, 1999, [2] Mislove et al., WSDM, 2010

Classic web and OSN content popularity



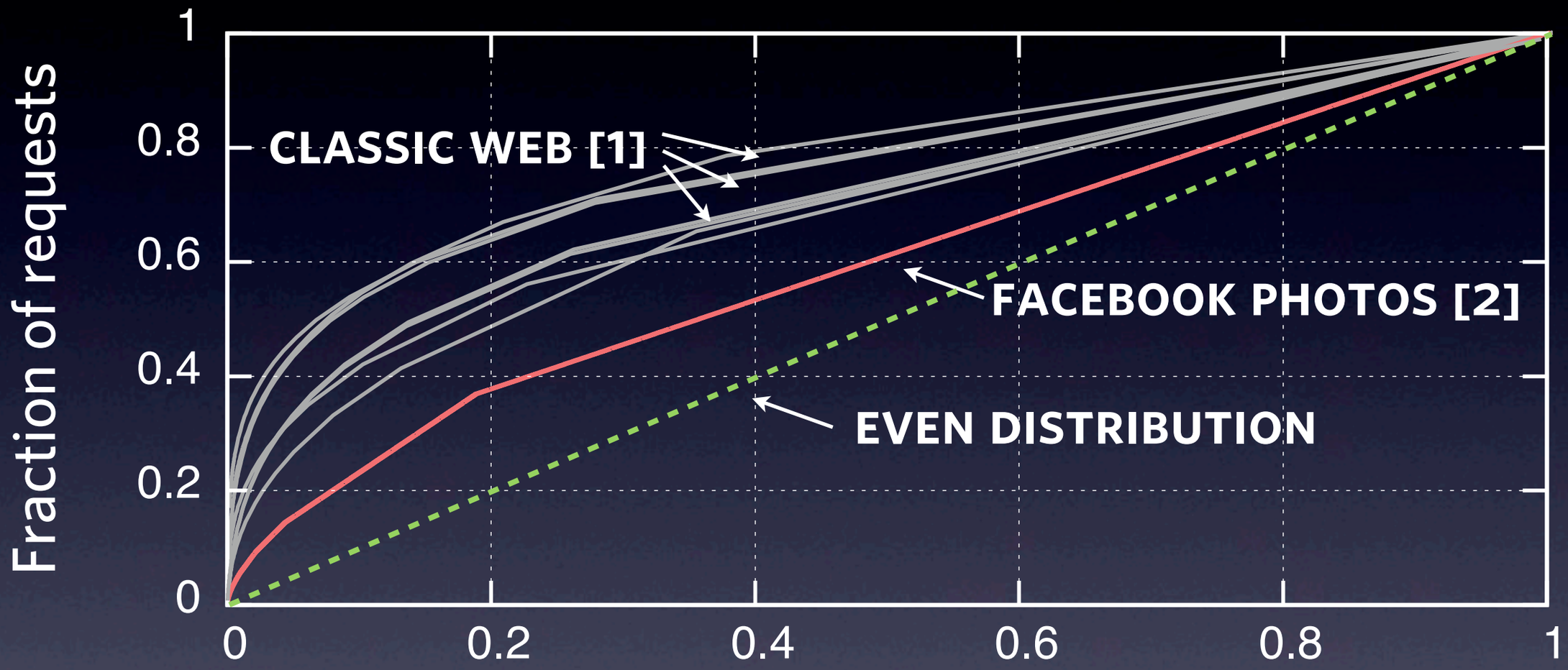
[1] Breslau et al., INFOCOM, 1999, [2] Mislove et al., WSDM, 2010

Classic web and OSN content popularity



[1] Breslau et al., INFOCOM, 1999, [2] Mislove et al., WSDM, 2010

Classic web and OSN content popularity



[1] Breslau et al., INFOCOM, 1999, [2] Mislove et al., WSDM, 2010

Implication: Caches/CDNs less effective

Popularity distribution much more even

- Objects have more narrow scope

In classic Web:

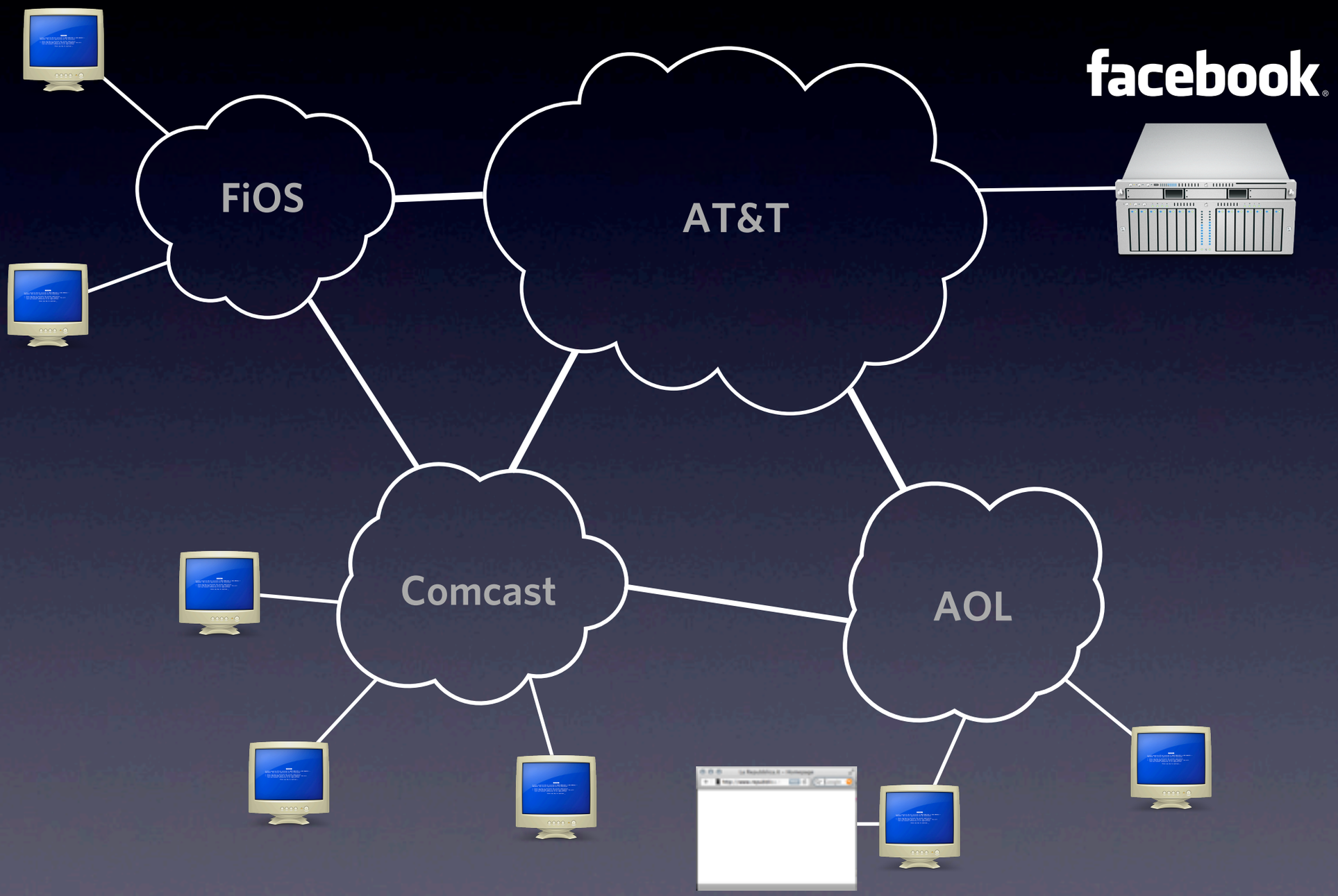
- Caching top 10% serves between 55% [1] and 95% [2] of requests
- Success of CDNs, web caches, …

In online social media:

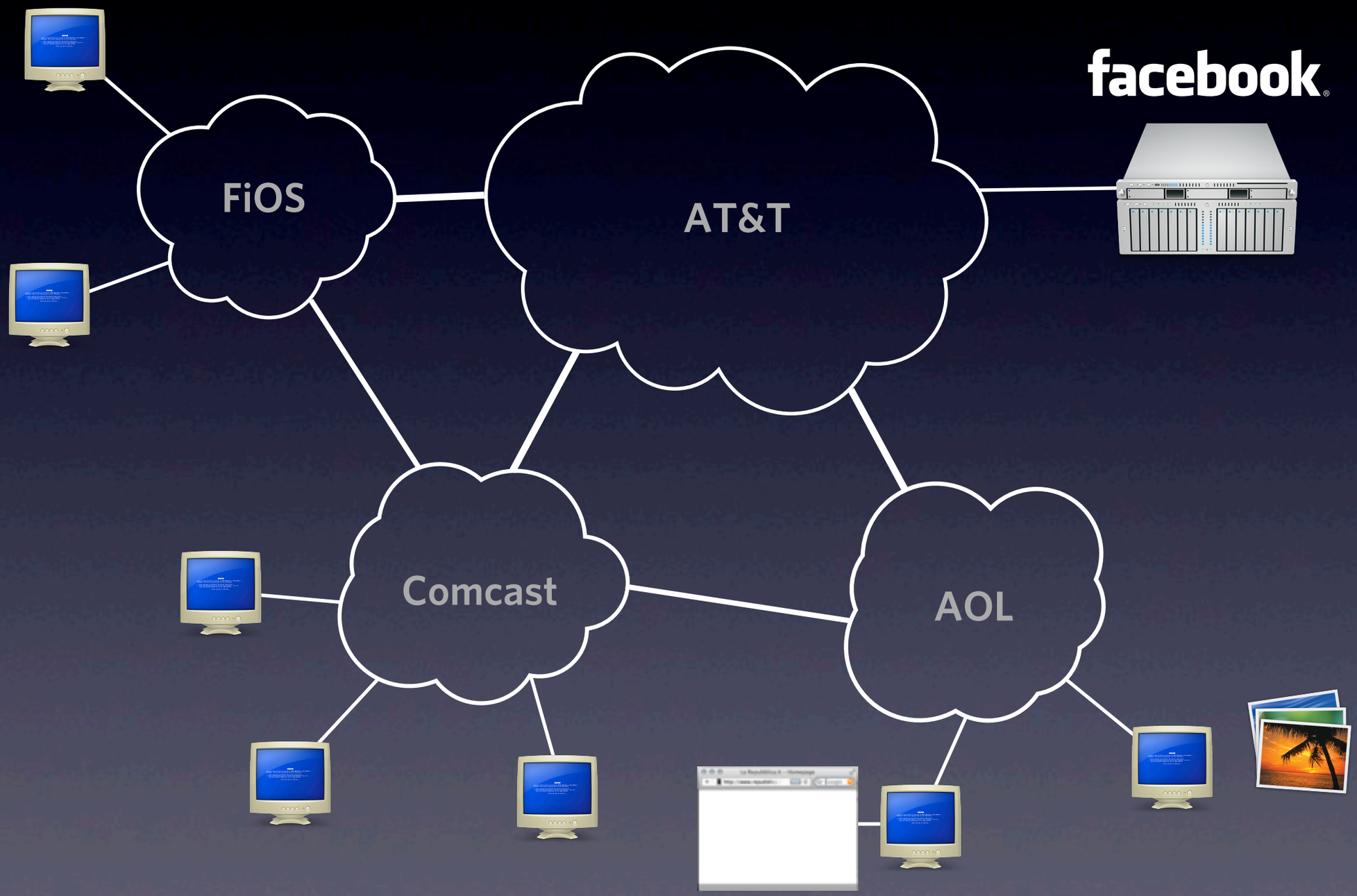
- Caching top 10% would only serve 27% [3] of requests

[1] Breslau et al., INFOCOM, 1999, [2] Arlitt et al. IEEE Network, 2000, [3] Mislove et al., WSDM, 2010

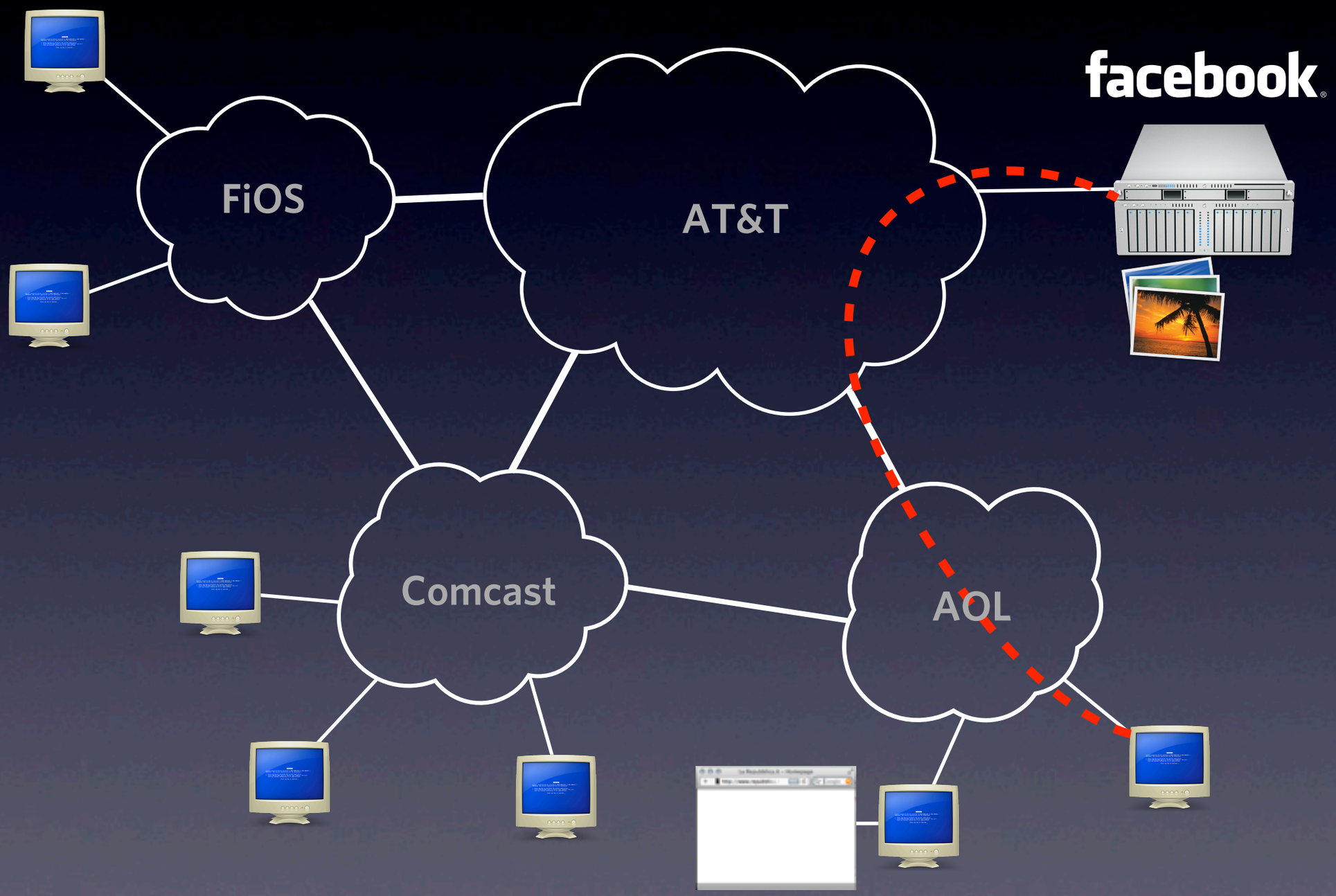
OSN content creation/exchange



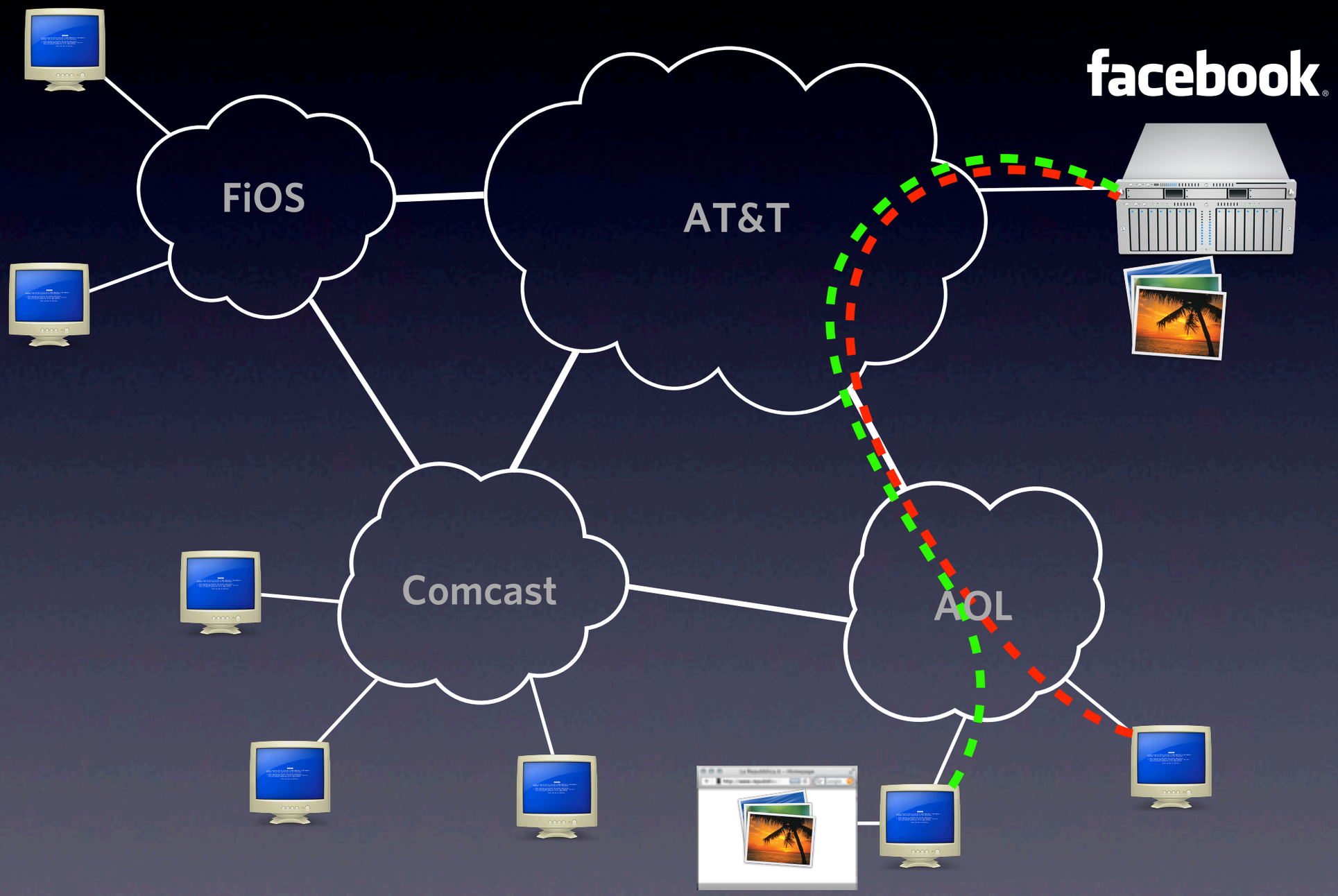
OSN content creation/exchange



OSN content creation/exchange



OSN content creation/exchange



Implication: Workload change

Significant content creation at network's edge

- Ease of digital content creation (photos, video)
- Ubiquity of Internet access (cell phone, iPad)

In Classic Web:

- Workload was "center-to-edge"
- Caching, CDNs take load off origin server

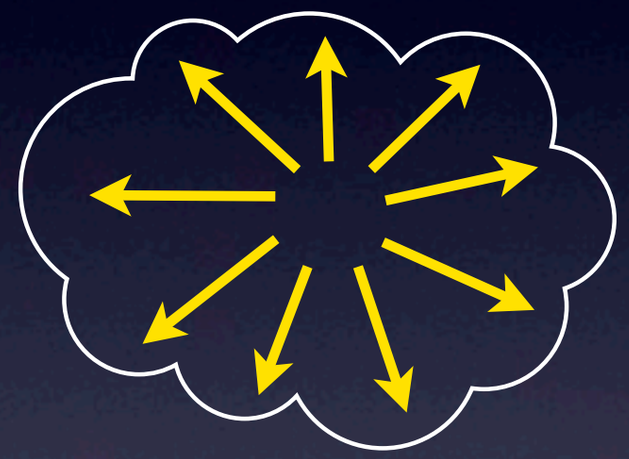
In Social Media:

- Workload is "edge-to-edge"
- Significant geographic locality

Implication: Workload change

Significant content creation at network's edge

- Ease of digital content creation (photos, video)
- Ubiquity of Internet access (cell phone, iPad)



In Classic Web:

- Workload was "center-to-edge"
- Caching, CDNs take load off origin server

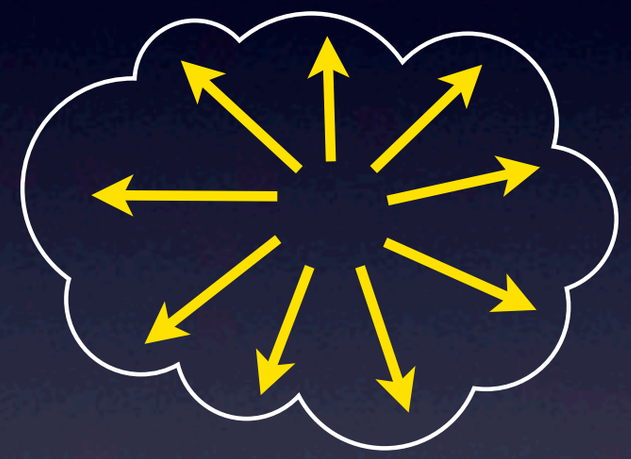
In Social Media:

- Workload is "edge-to-edge"
- Significant geographic locality

Implication: Workload change

Significant content creation at network's edge

- Ease of digital content creation (photos, video)
- Ubiquity of Internet access (cell phone, iPad)

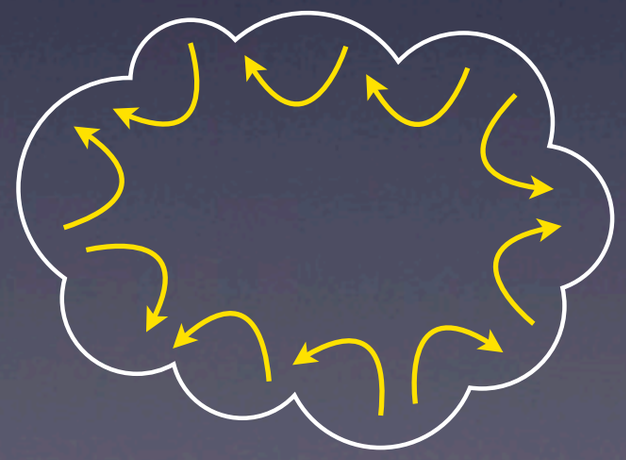


In Classic Web:

- Workload was "center-to-edge"
- Caching, CDNs take load off origin server

In Social Media:

- Workload is "edge-to-edge"
- Significant geographic locality



How is OSN content being delivered?

Web 1.0 “centralized” architectures dominate

- Akamai, Limelight, Clearway, …
- Facebook serves much of its own content

Mismatch between infrastructure, workload

What are the current solutions?

Alternate Approaches

Current Solutions

- Decentralized CDN
  - Coral
  - Not self-sustaining in the long run

- User participating in CDNs
  - Akamai's NetSession, FireCoral
  - Require additional software

- Decentralized social network systems
  - PeerSoN, Diaspora
  - Small user base

This talk

Goal: Build content distribution system for OSNs

Keep content exchange at the edge

Requirements

Works with today's web sites

No client side changes

facebook

WebCloud

Recruit user browsers to serve content

This talk

Goal: Build content distribution system for OSNs

Keep content exchange at the edge

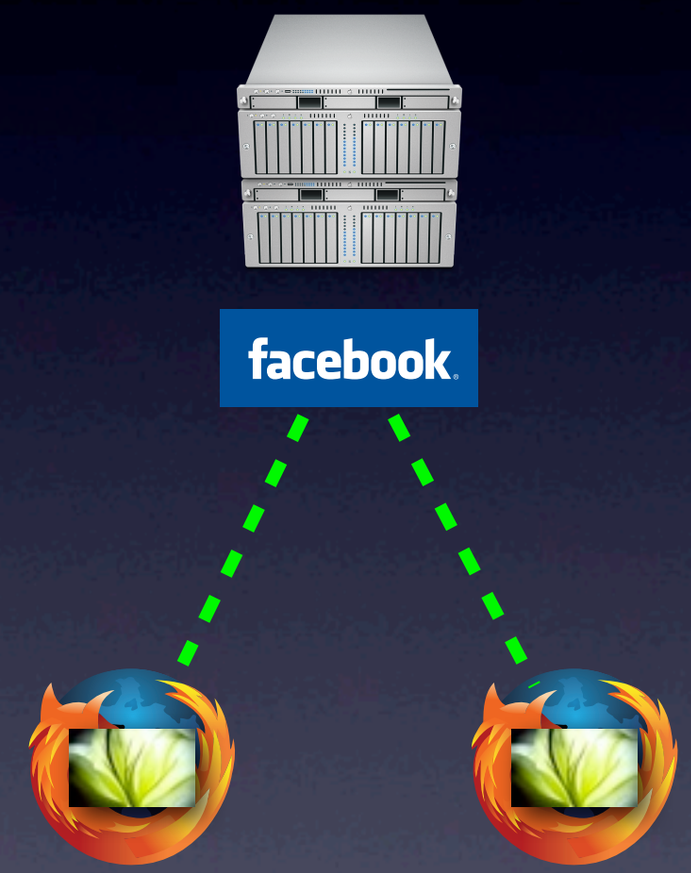
Requirements

Works with today's web sites

No client side changes

WebCloud

Recruit user browsers to serve content



This talk

Goal: Build content distribution system for OSNs

Keep content exchange at the edge

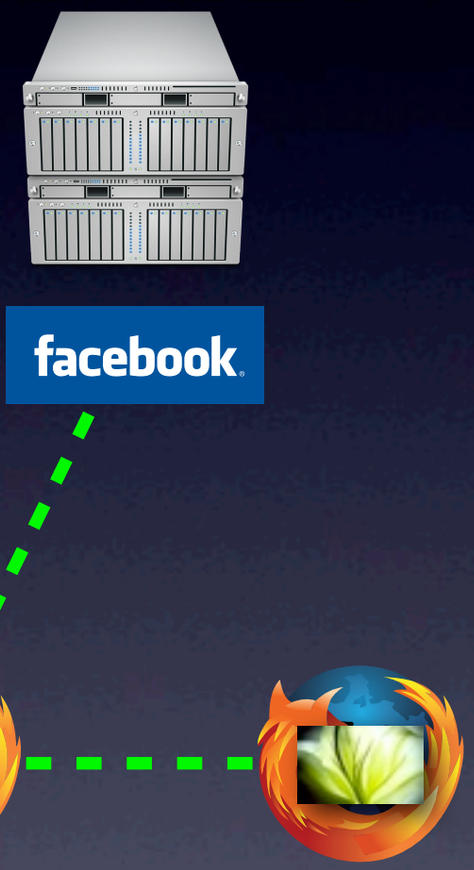
Requirements

Works with today's web sites

No client side changes

WebCloud

Recruit user browsers to serve content



Outline

1. ~~Motivation~~

2. WebCloud design

3. Evaluation

WebCloud design challenges

WebCloud: Drop-in content distribution system for OSNs

- Serves as a distributed cache
- Assume content always available from OSN

Want to make WebCloud work with today's sites, browsers

- Reason: Users unlikely install software

Key challenge: Browsers not designed to communicate directly

- Browsers distinct from Web servers
- Use novel techniques to allow browser to serve content

Redirector proxy: the middlebox

Introduce a redirector proxy

- Allow browsers to "talk" to other browsers

Place *redirector proxies* in each ISP/region

- Like Akamai server, but doesn't store any content
- Maintain open connections to online web visitors
- Run by OSN provider

Keeps track of content in each user's browser

- Serves as a directory for content

What do we need in browsers?

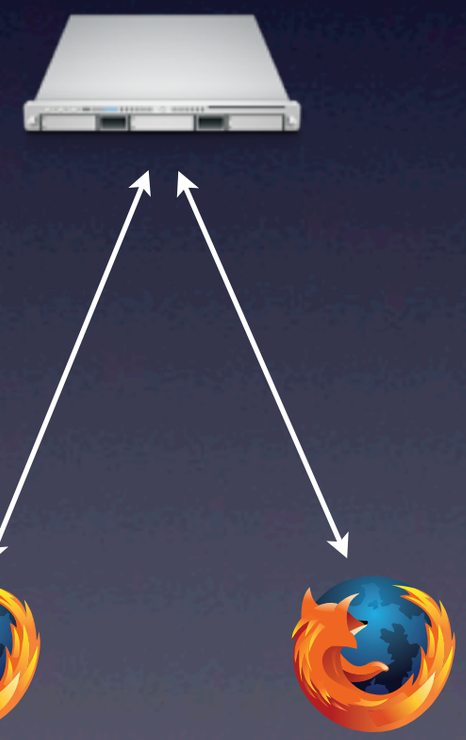
Redirector proxy: the middlebox

Introduce a redirector proxy

- Allow browsers to "talk" to other browsers

Place *redirector proxies* in each ISP/region

- Like Akamai server, but doesn't store any content
- Maintain open connections to online web visitors
- Run by OSN provider



Keeps track of content in each user's browser

- Serves as a directory for content

What do we need in browsers?

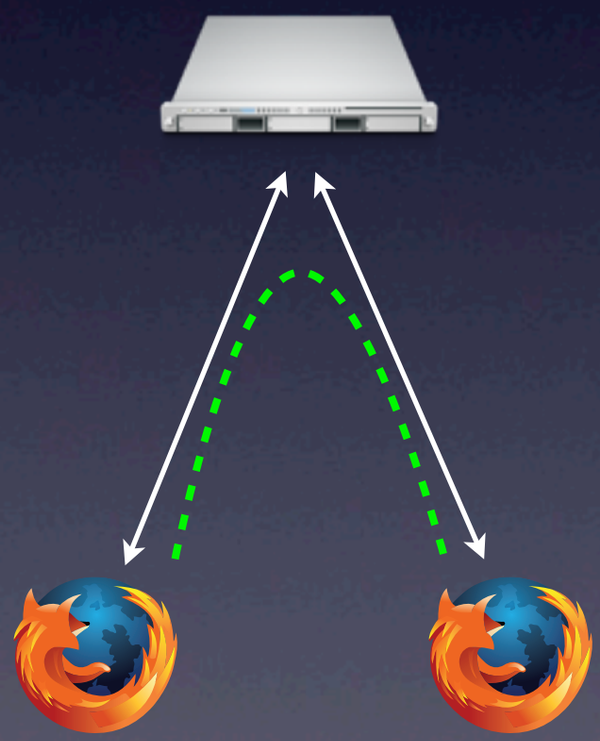
Redirector proxy: the middlebox

Introduce a redirector proxy

- Allow browsers to "talk" to other browsers

Place *redirector proxies* in each ISP/region

- Like Akamai server, but doesn't store any content
- Maintain open connections to online web visitors
- Run by OSN provider



Keeps track of content in each user's browser

- Serves as a directory for content

What do we need in browsers?

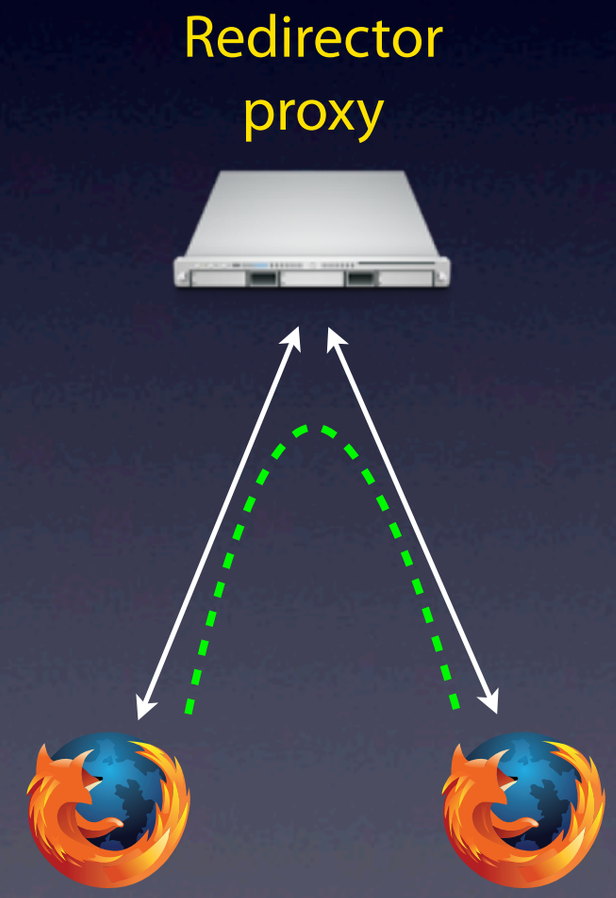
Redirector proxy: the middlebox

Introduce a redirector proxy

- Allow browsers to "talk" to other browsers

Place *redirector proxies* in each ISP/region

- Like Akamai server, but doesn't store any content
- Maintain open connections to online web visitors
- Run by OSN provider



Keeps track of content in each user's browser

- Serves as a directory for content

What do we need in browsers?

Client side changes

Implement WebCloud in Javascript

- Add it to the site's pages

Browsers use WebSockets/XHR to communicate with middlebox

- Allows bi-directional communication
- Online client is always connected to redirect proxy

+

Use LocalStorage to storage browsed content

- Persistent cache, up to 5MB/site
- Easily programmatically accessed

Insert downloaded objects in LocalStorage

- Treated like LRU cache

Using WebCloud

Provide JavaScript library for sites to use WebCloud

1. Include WebCloud Javascript

```
<script src=”webcloud.js”>
```

2. Change mechanism for loading content

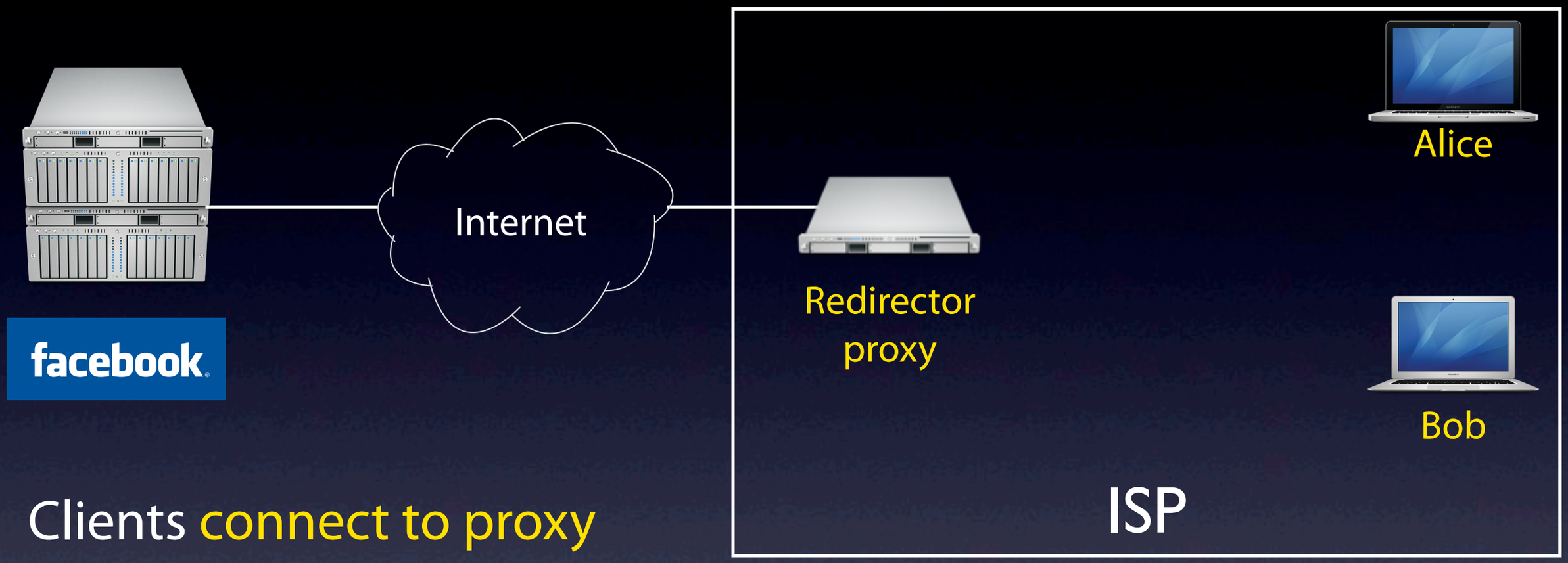
WebCloud content referred to by content-hash

```
<img id="pic-id" src=”http://www.foo.com/...”/>
```

replaced with

```
<img id="pic-id"/>
<script>
  webcloud.load("pic-hash", "pic-id");
</script>
```

Putting it all together



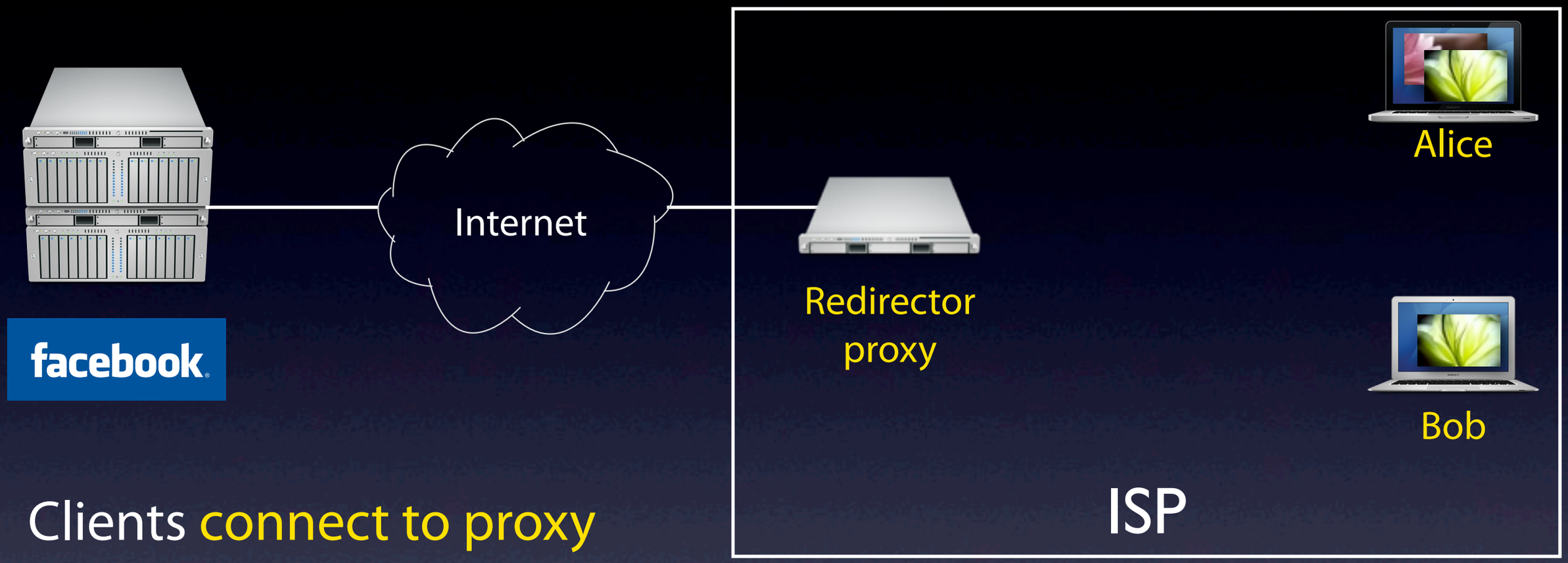
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Putting it all together



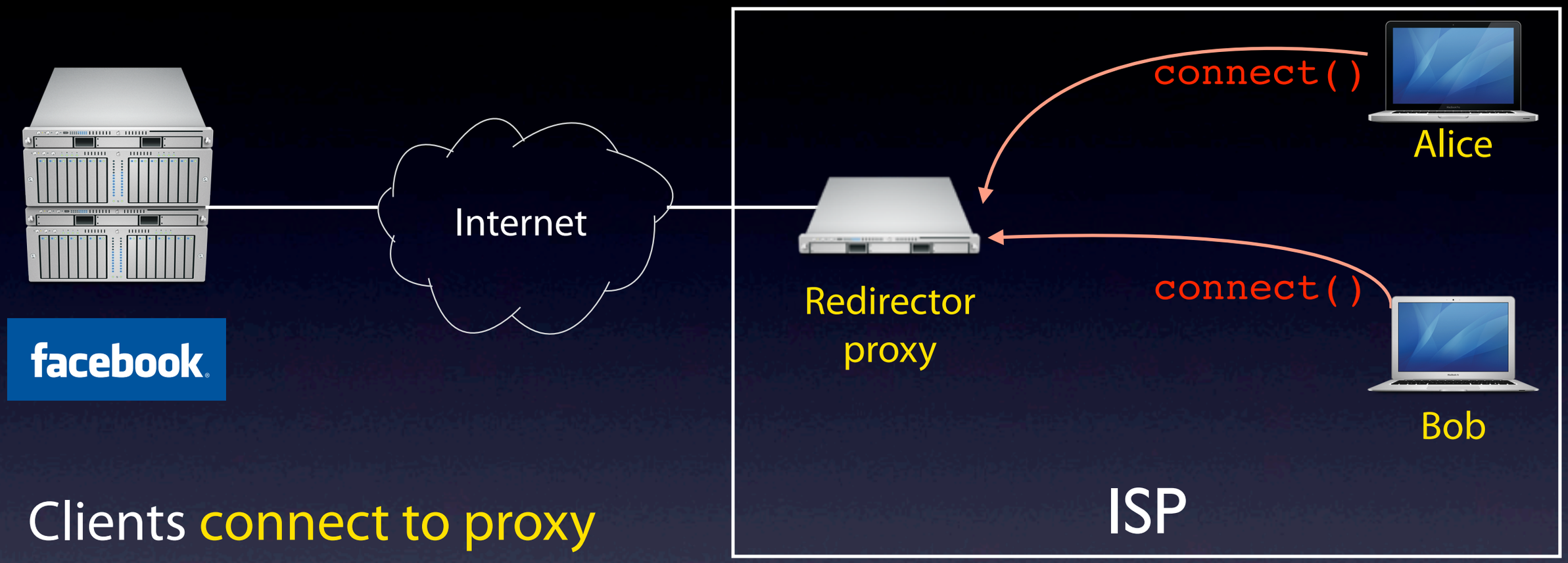
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Putting it all together



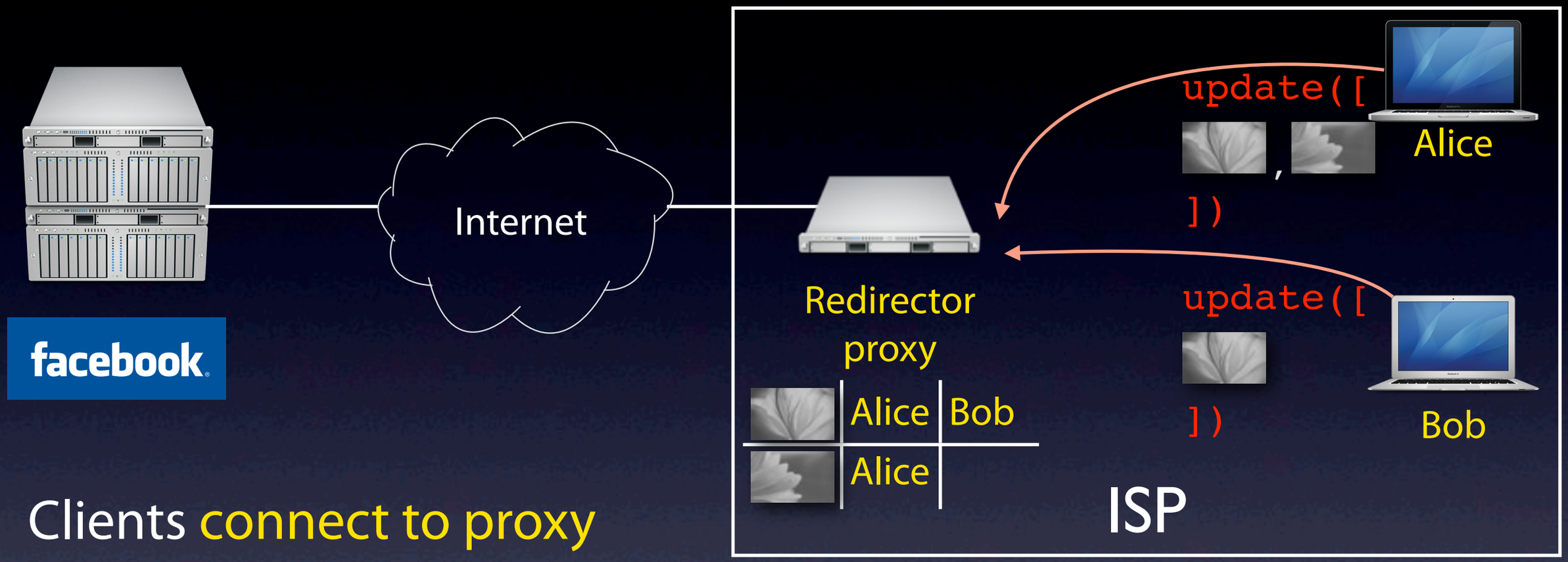
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Putting it all together



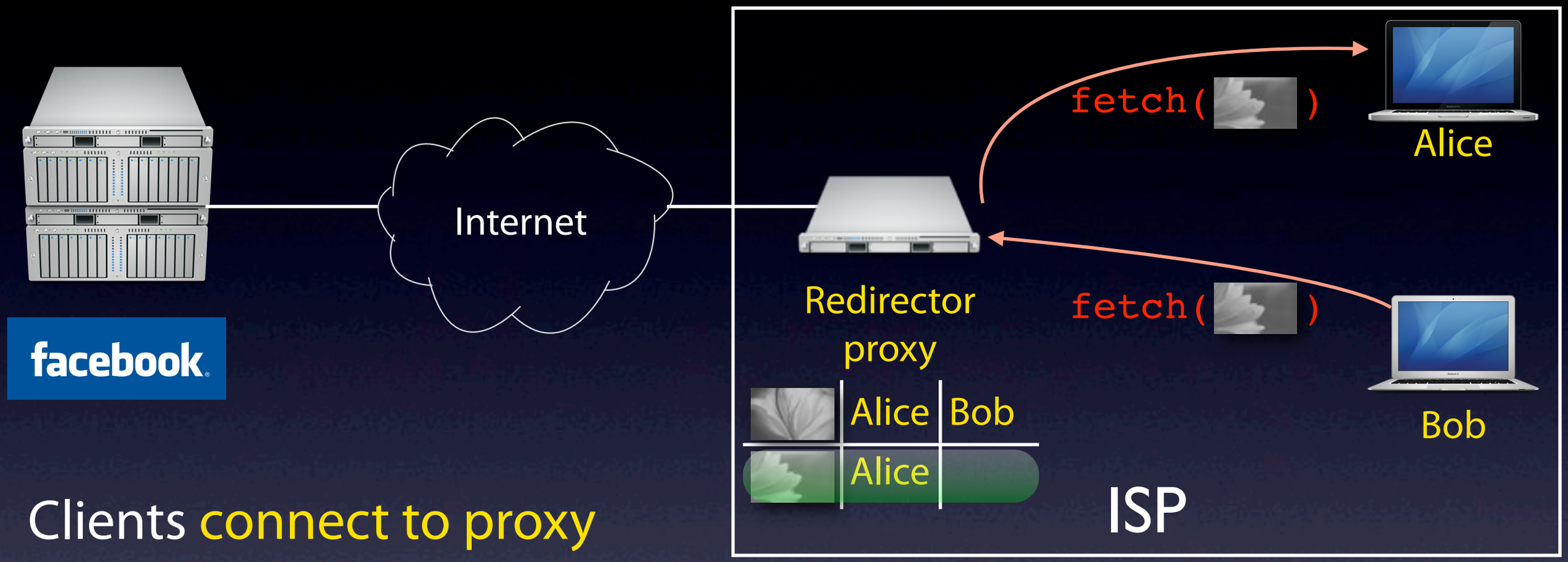
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Putting it all together



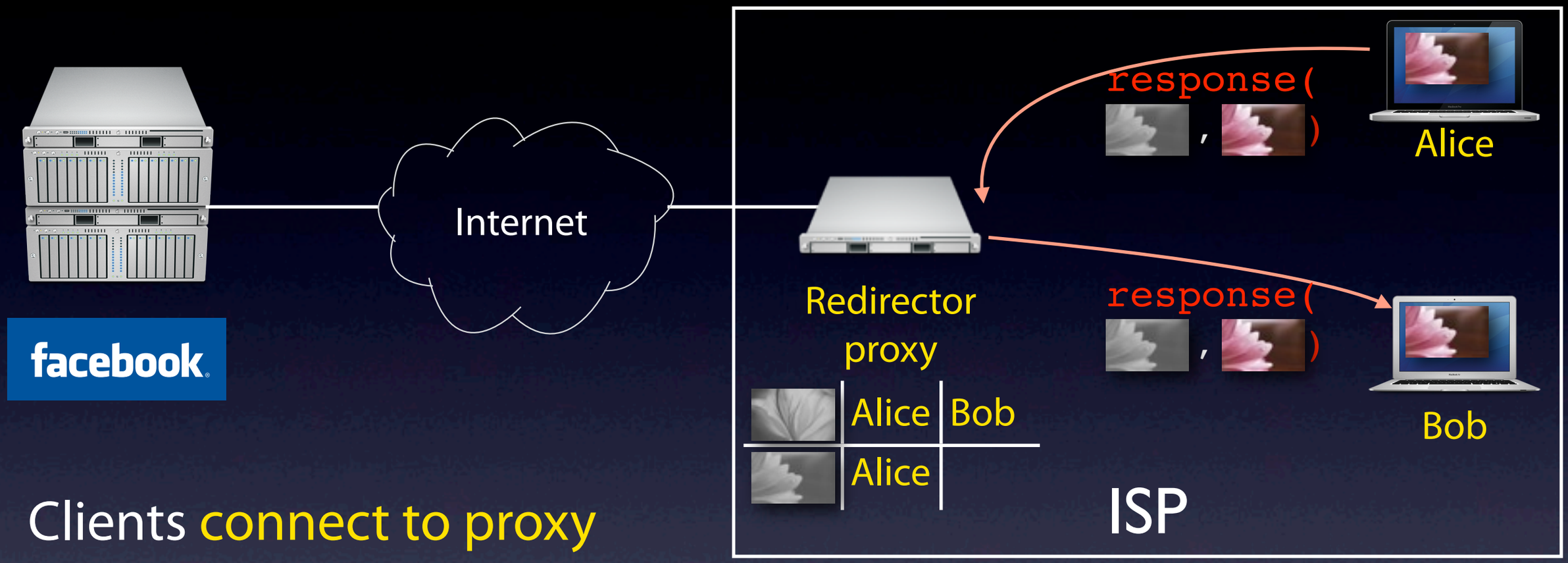
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Putting it all together



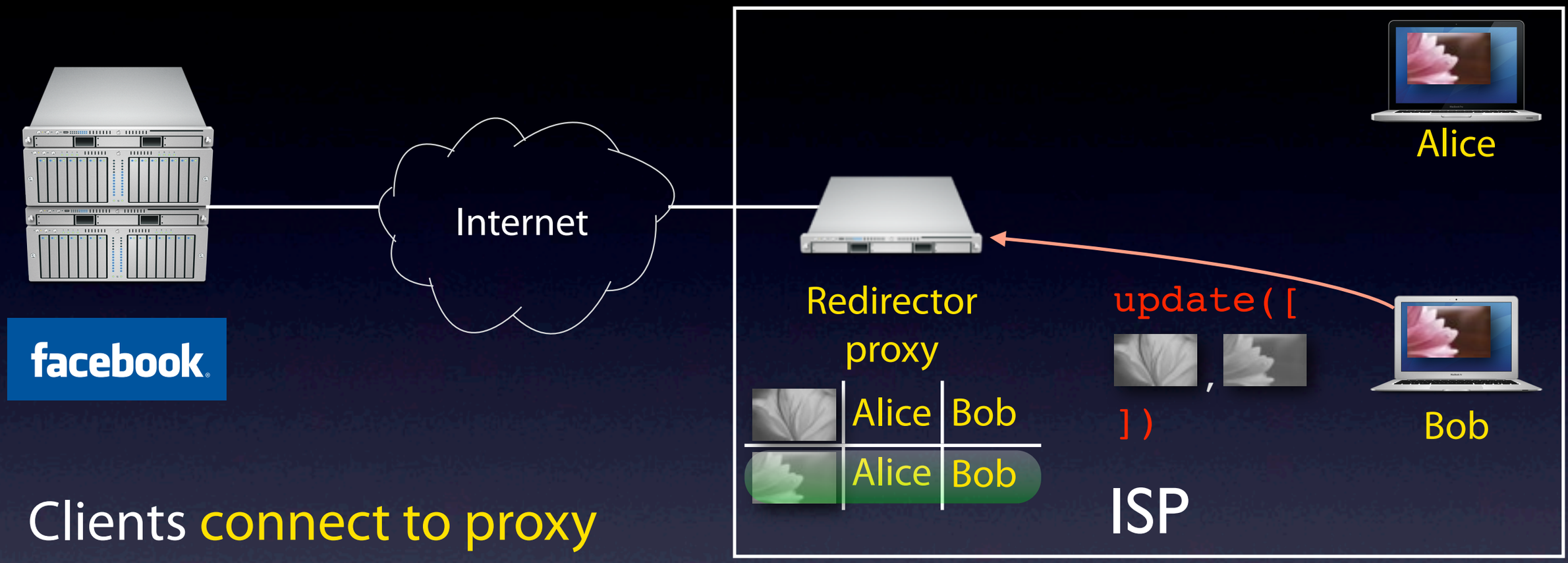
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Putting it all together



Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Security and Privacy

Can users serve forged content?

- Detected forged content using content hash

Can users view content they are not allowed to?

- Content secure by its hash
- * Same semantics as Facebook and other sites today

Can users figure out what others have browsed?

- *k*-anonymity, k is the number of online users
  - connected to the same proxy who are able to view the photo

Content viewable to a small set of users

- Disable loading via WebCloud
- Cover traffic

Outline

~~1. Motivation~~

~~2. WebCloud Design~~

3. Evaluation

Evaluation overview

Focus on serving Facebook photos

- Most popular application on Facebook
- Easy to get data, users

Implemented WebCloud proxy and client-side Javascript

- Client: 1,226 lines of Javascript
- Proxy: 1,283 lines of Python

Want to answer these questions

- Is there extra latency/overhead? -- Micro benchmark
- What WebCloud hit rate can we expect? -- Simulation
- Does it work with today's browsers/sites? -- Real deployment

Is there additional latency?

Accessed from	Served from		
	Facebook (today)	WebCloud	
		LAN	Cable
LAN	668 ms	63 ms	398 ms
Cable	690 ms	153 ms	532 ms

No, in fact, always faster than getting from Facebook

- All simulations ran in Boston, like deployment
- Loading 62KB photos

Approximate 60 ms latency for fetching from browser

What WebCloud hit rate can we expect?

Simulate large-scale deployment using crawled Facebook data

- New Orleans network, 63K users, 1.8M links
- 1.07M comments on 816K photos

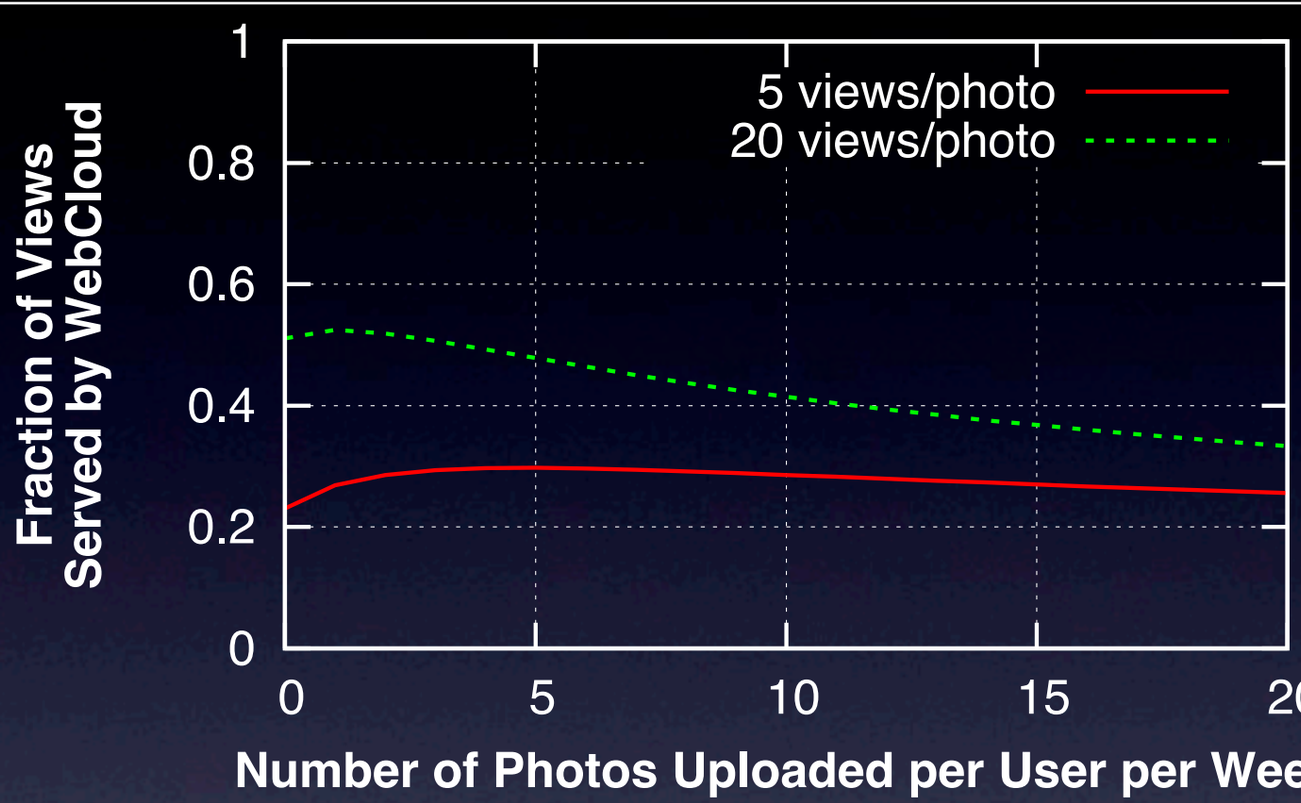
Why synthetic data?

- No public available data on Facebook user
  - online/offline trace
  - photo viewing behavior

Simulate 1-week WebCloud deployment

- Many different configurations; more in paper

What WebCloud hit rate can we expect?

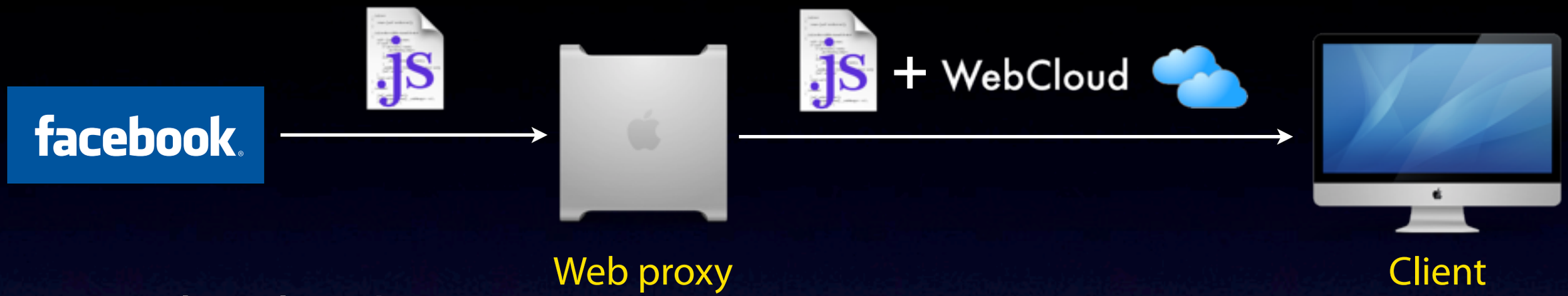


Varying the number of uploaded photos

- Average number of views per photo is 5 or 20
- Hit-rate increases then drops off, due to fixed cache size

Overall hit-rate range between 23% and 57%

Real-world deployment



Set up local web proxy

- Injects WebCloud Javascript into Facebook's pages
- Emulates Facebook deploying WebCloud for photos

Deployed WebCloud to real users

- 17 users for 10 days
- Total of 2,069 photos viewed, 26% served from WebCloud

Works with Firefox, Safari, Chrome

- Average browser could store 56 photos

Summary

Compared OSN and classic web traffic

- OSN workload substantially different

But, still using centralized delivery architectures

- Affecting ability to serve new, rich content

WebCloud: First step towards decentralized Web content delivery

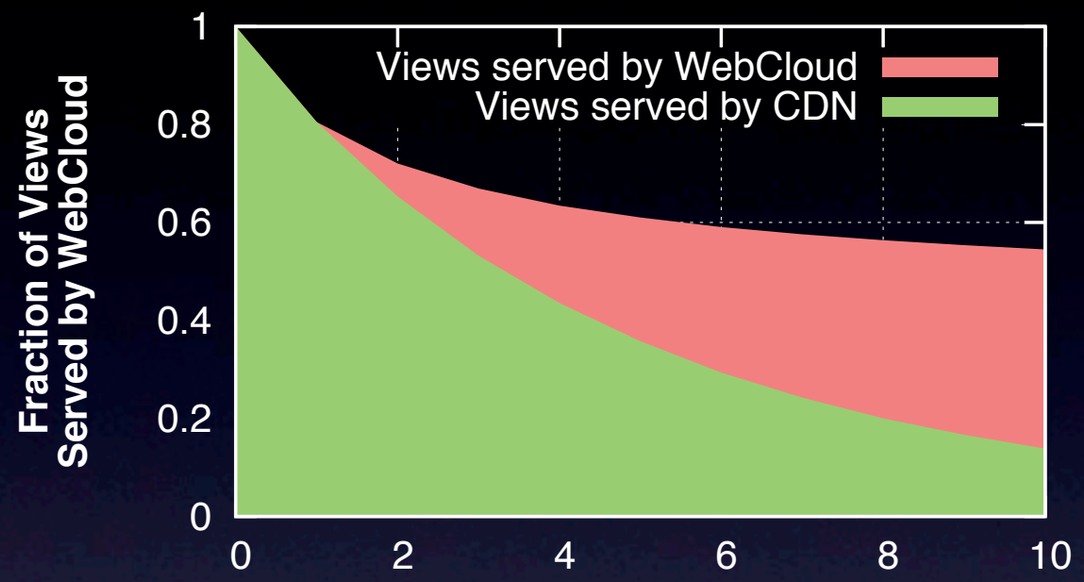
- Users help serve content they create
- Implemented using existing browser features

Evaluation demonstrated practicality, efficacy

Thank you

Questions?

How does WebCloud compare to CDN?



CDN configuration

- Store content that has been requested *k* times
- Unlimited storage

WebCloud benefits

- Serves over 25% when k = 5
- Serves over 40% when k = 10

Mobile Devices

Works on Mobile browsers in Android and iOS devices

- Short session time
- Only work when active

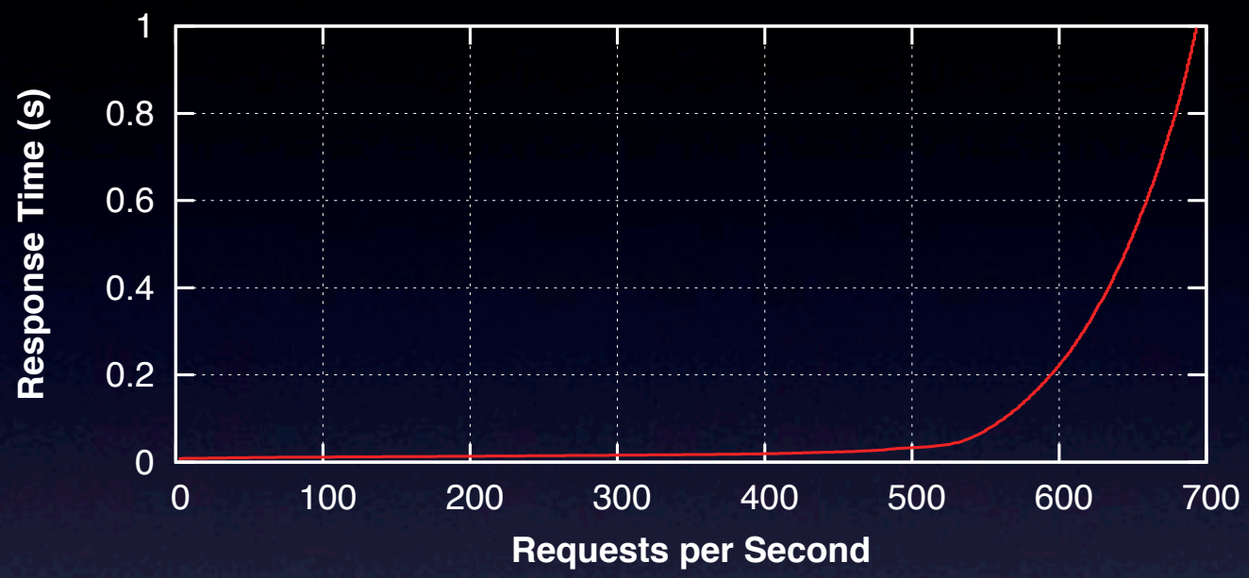
Site-specific Apps (e.g. Facebook for iOS)

- Background service, keep connection with redirector proxy

Evaluation of mobile WebCloud

- iOS 4.2 background VOIP app
- Serve a 60 KB photo every 5 sec
- 5031 requests over 8.26 hours via 3G
- 24700 requests over 34.9 hours via 3G
- Bandwidth usage, 72 MB max, 2 MB average

Is the server scalable?



Server configuration

- 4-core 2.83 GHz machine
- 16 GB of RAM

500 `fetch` requests per second

- Under 10 ms of latency
- Support over 30,000 online users

Resources Slides

:)

Content is created at the edge

In traditional Web:

- Workload was "center-to-edge"
- Caching, CDNs take load off origin server

Content is created at the edge

In traditional Web:

- Workload was "center-to-edge"
- Caching, CDNs take load off origin server

Content is created at the edge

In traditional Web:

- Workload was "center-to-edge"
- Caching, CDNs take load off origin server

Content is created at the edge

In traditional Web:

- Workload was "center-to-edge"
- Caching, CDNs take load off origin server

In online social media:

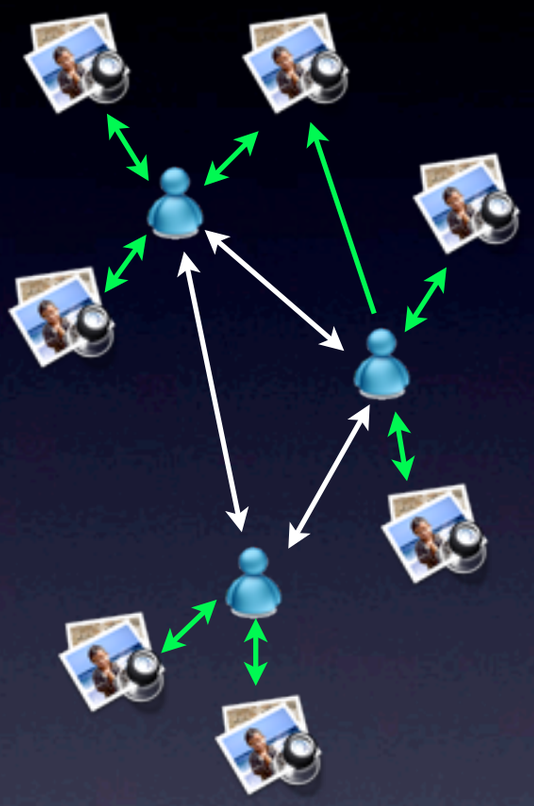
- Significant content creation at network's edge
  - Ease of digital content creation (photos, video)
  - Ubiquity of Internet access (cell phone, iPad)
- Workload is "edge-to-edge"
  - Significant geographic locality [1]

So, what's the problem?

Content is created at the edge

In traditional Web:

- Workload was "center-to-edge"
- Caching, CDNs take load off origin server



In online social media:

- Significant content creation at network's edge
  - Ease of digital content creation (photos, video)
  - Ubiquity of Internet access (cell phone, iPad)
- Workload is "edge-to-edge"
  - Significant geographic locality [1]

So, what's the problem?

This talk

Goal: Move towards more decentralized content exchange

- Keep content exchange at the edge

Requirement

- Works with today's web sites

facebook

Idea

- Serves content from user browsers

This talk

Goal: Move towards more decentralized content exchange

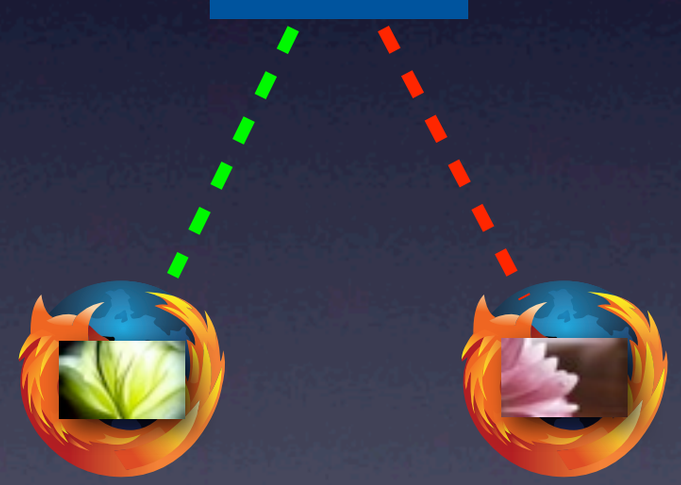
Keep content exchange at the edge

Requirement

Works with today's web sites

Idea

Serves content from user browsers



This talk

Goal: Move towards more decentralized content exchange

- Keep content exchange at the edge

Requirement

- Works with today's web sites

WebCloud

Idea

Serves content from user browsers

Security and Privacy

Can WebCloud serve forged content?

- Detected forged content using content hash

Can users view content they are not allowed to?

- Content secure by its hash
  - * Same semantics as Facebook and other sites today

Perform a denial-of-service (DOS) attack on the redirector proxy

- Block accounts, IP addresses, or subnets

Privacy

Only allows users to fetch content that they could access

- Cannot view content they could not
- Forbid disclosing content to unauthorized third party

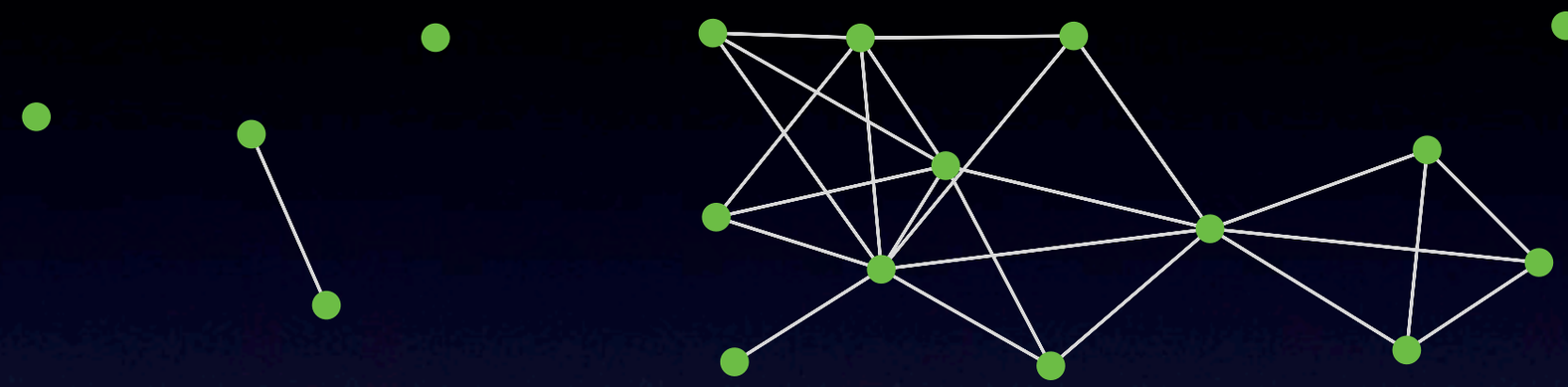
Can users figure out what others have browsed?

- *k*-anonymity, k is the number of online users
  - connected to the same proxy who are able to view the photo

Content viewable to a small set of users

- Disable loading via WebCloud
- Cover traffic

Does it work with today's browsers?



Deployed WebCloud within Northeastern CS College

17 users for 10 days

Total of 2,069 photos viewed

26% served from WebCloud

Works with Firefox, Safari, Chrome

Average browser could store 56 photos

WebCloud design overview

WebCloud 

First step towards decentralized Web content delivery

- Challenge: Web doesn't support decentralization
- Browsers distinct from Web servers

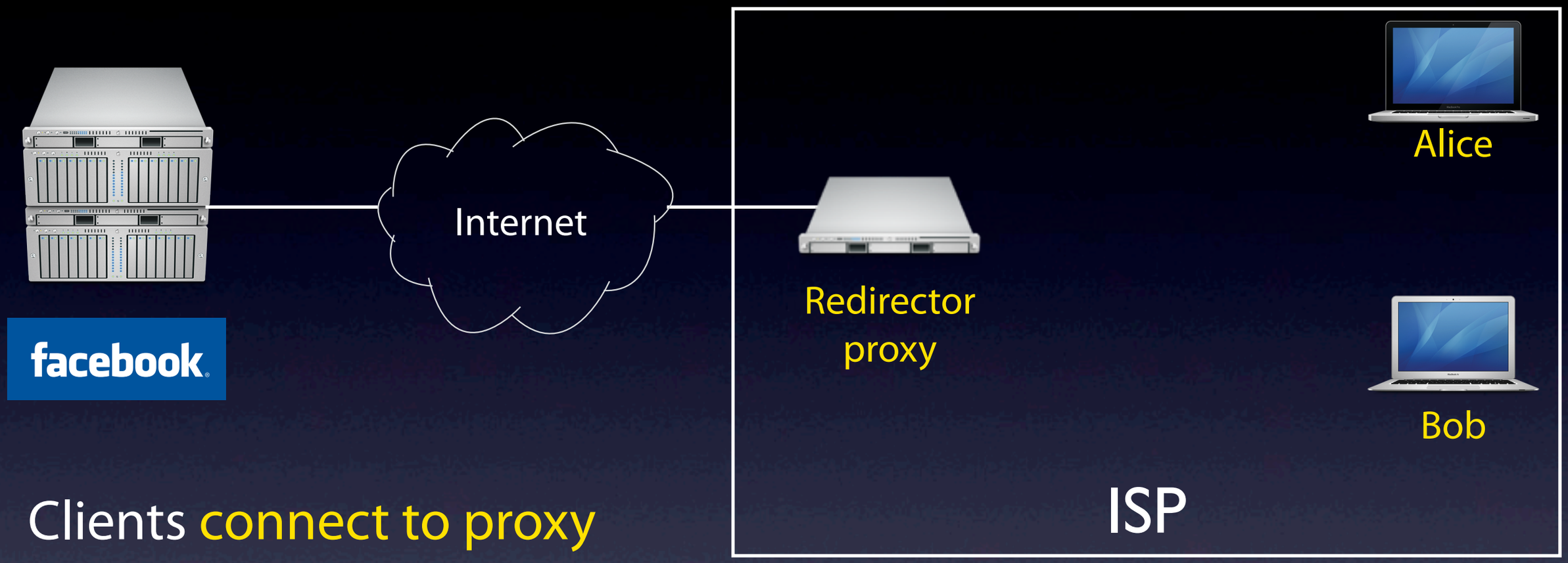
Use novel techniques to allow browser to serve content

- No client-side changes
- Users help serve content they upload
- Result: Scalable, workload-matching architecture

WebCloud is designed to

- be deployed by a web site, e.g. Facebook
- be compatible with today's web browsers
- serve as a cache for content shared between users

Distributed cache



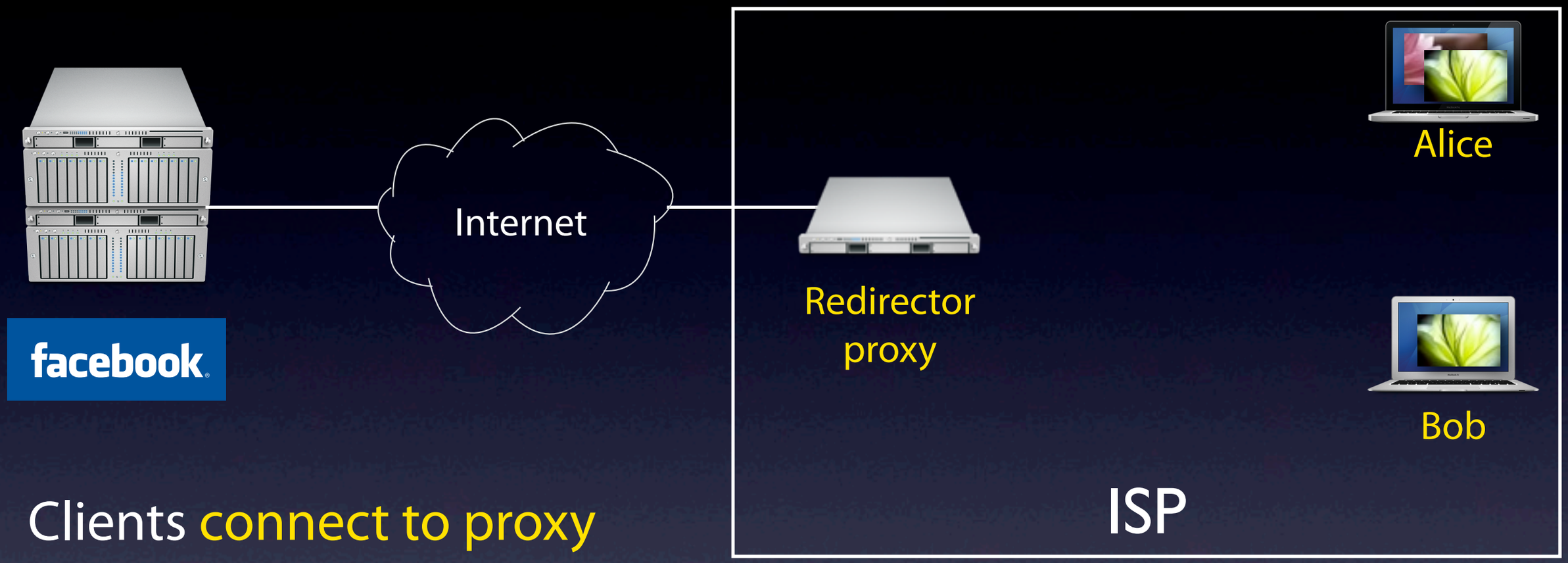
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



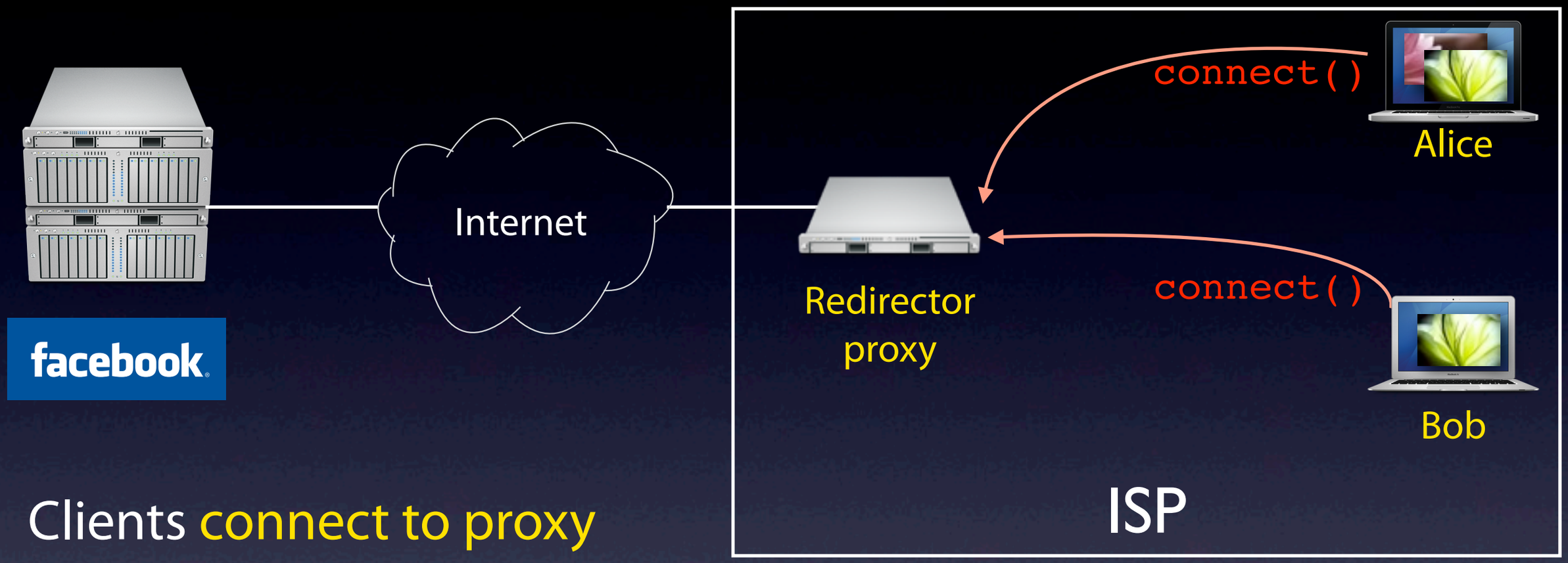
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



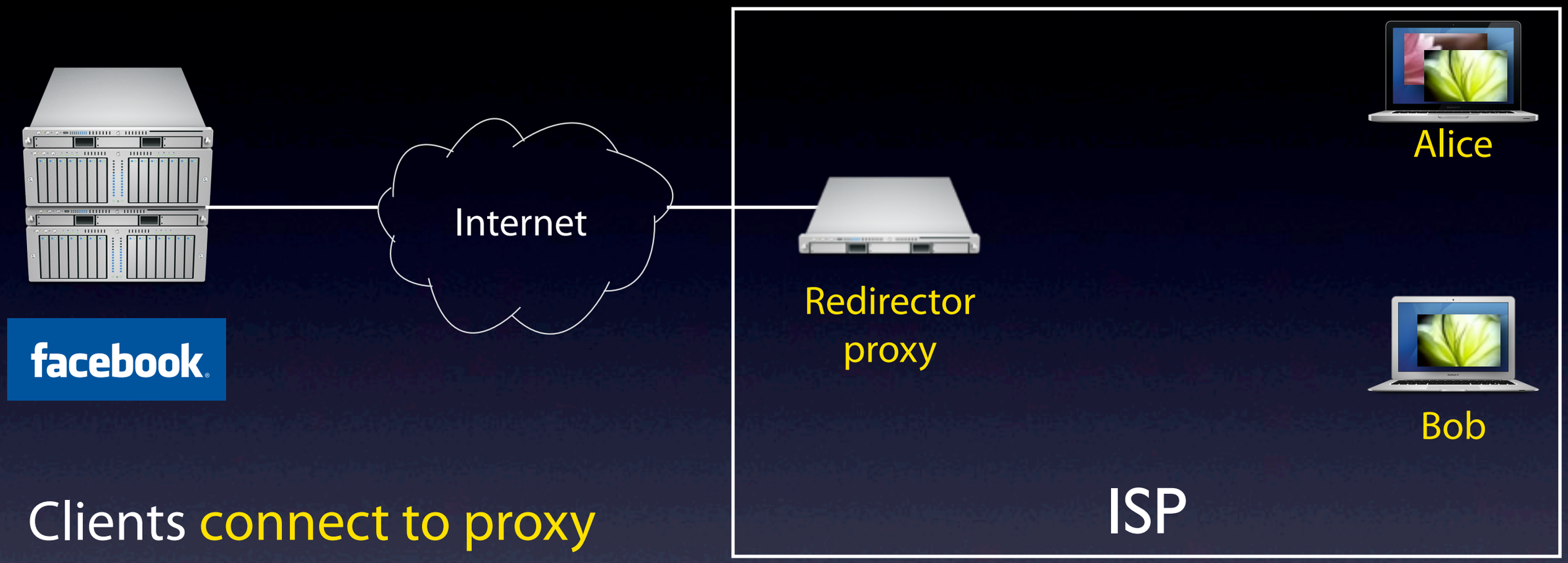
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



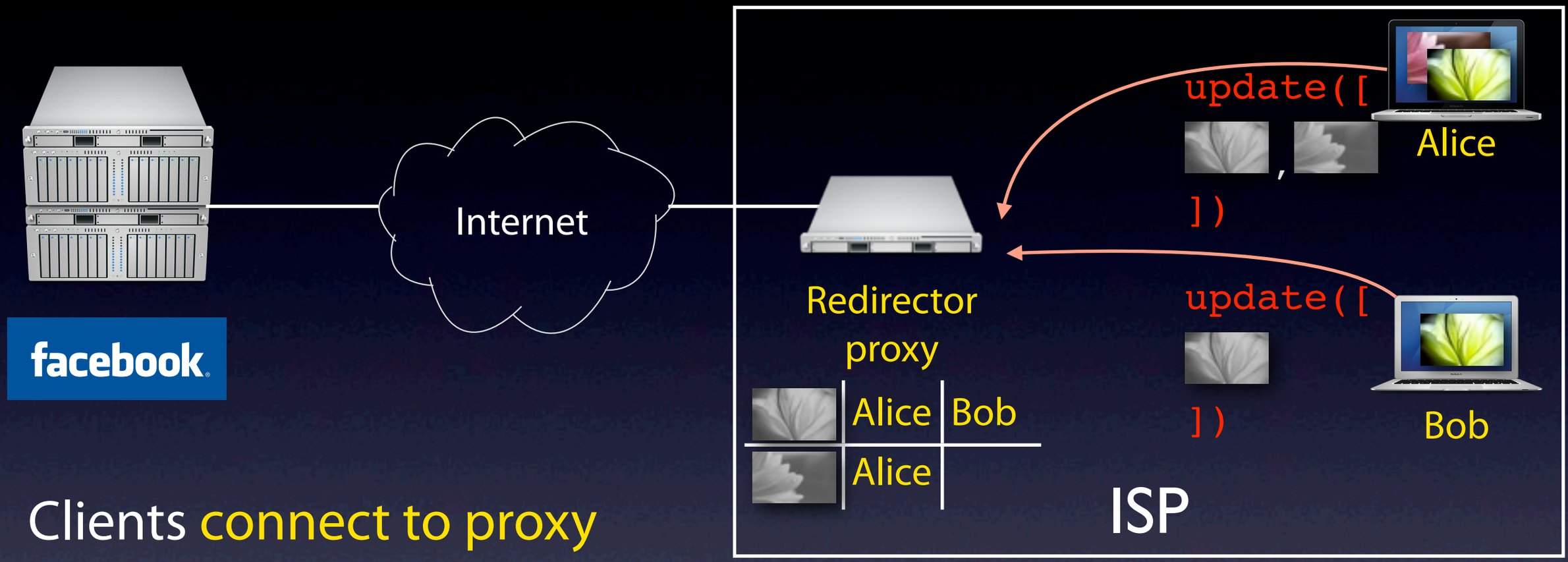
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



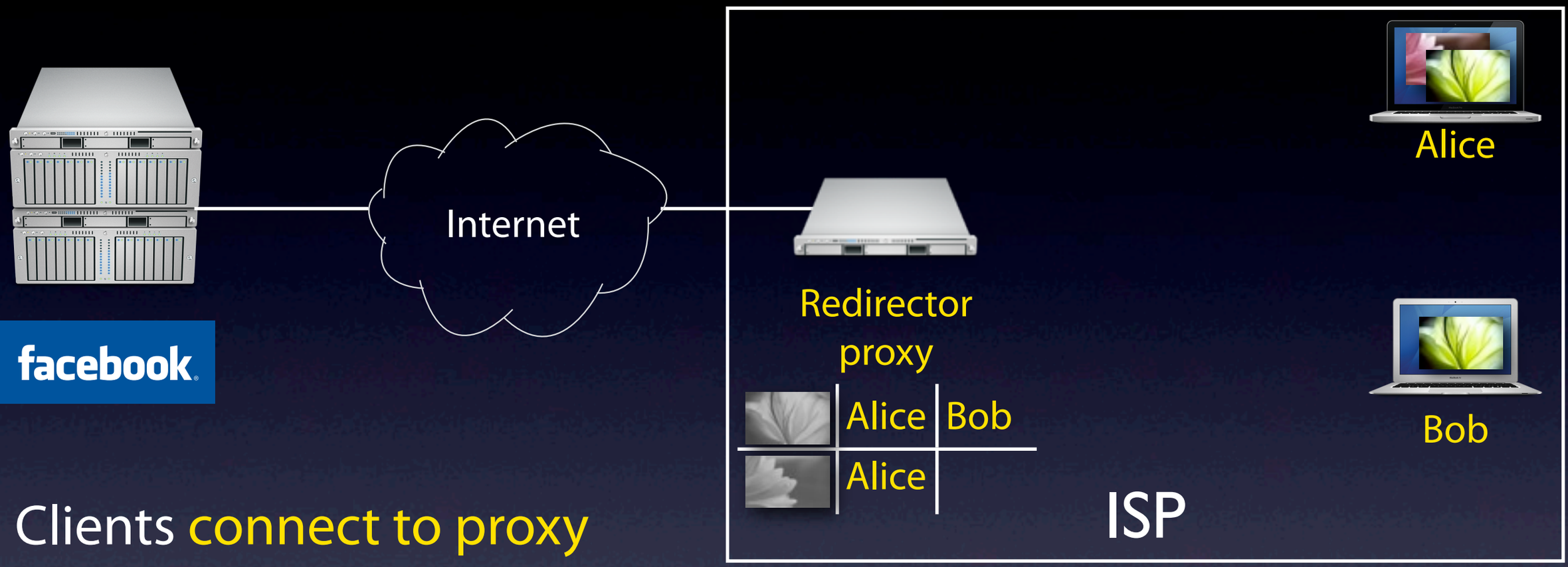
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



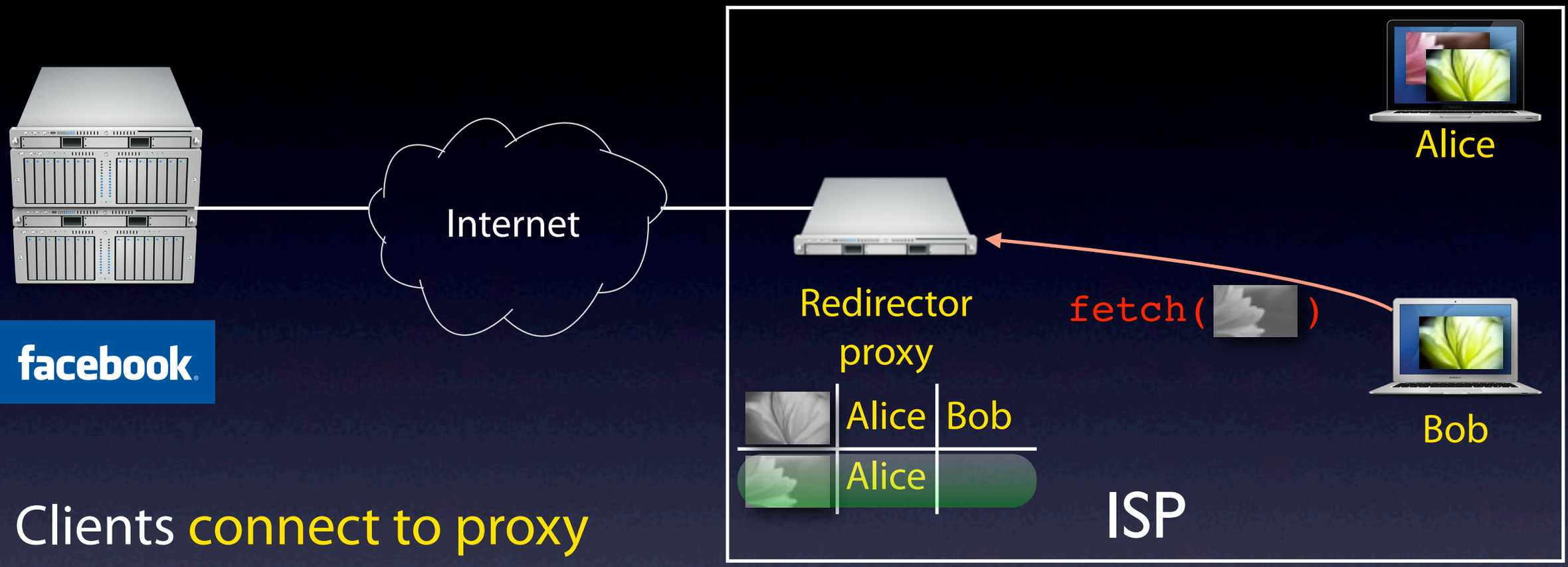
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



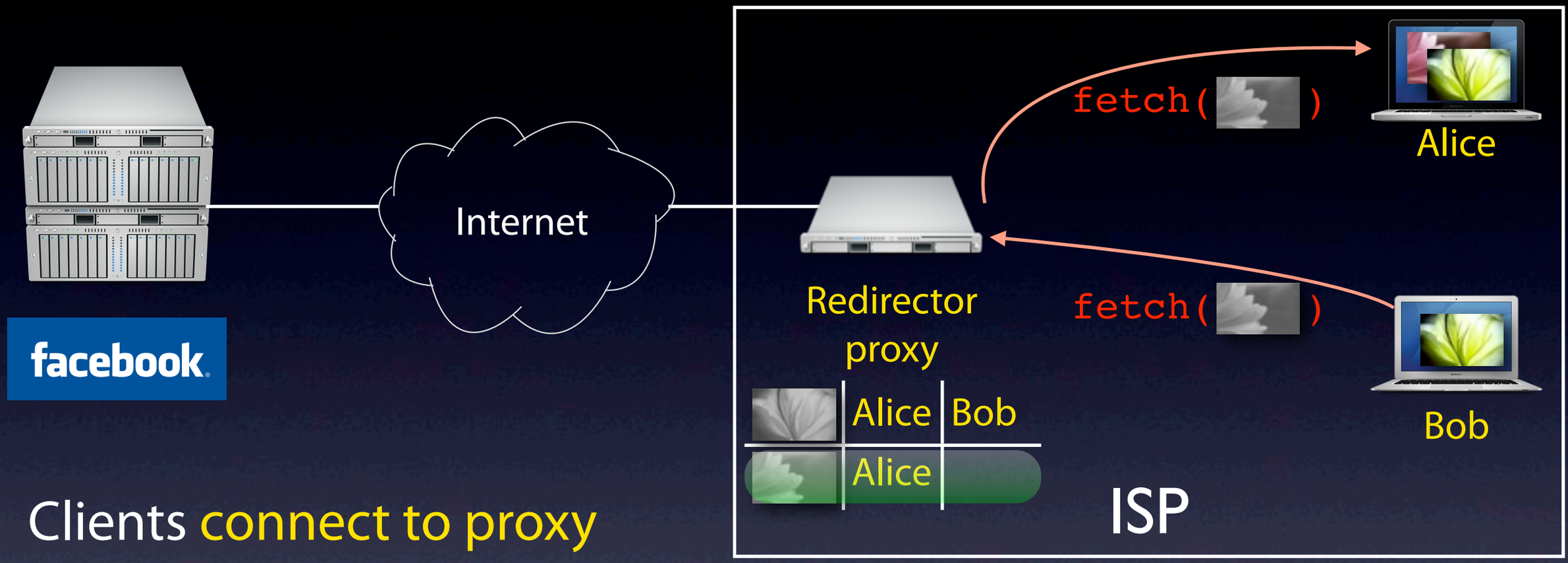
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



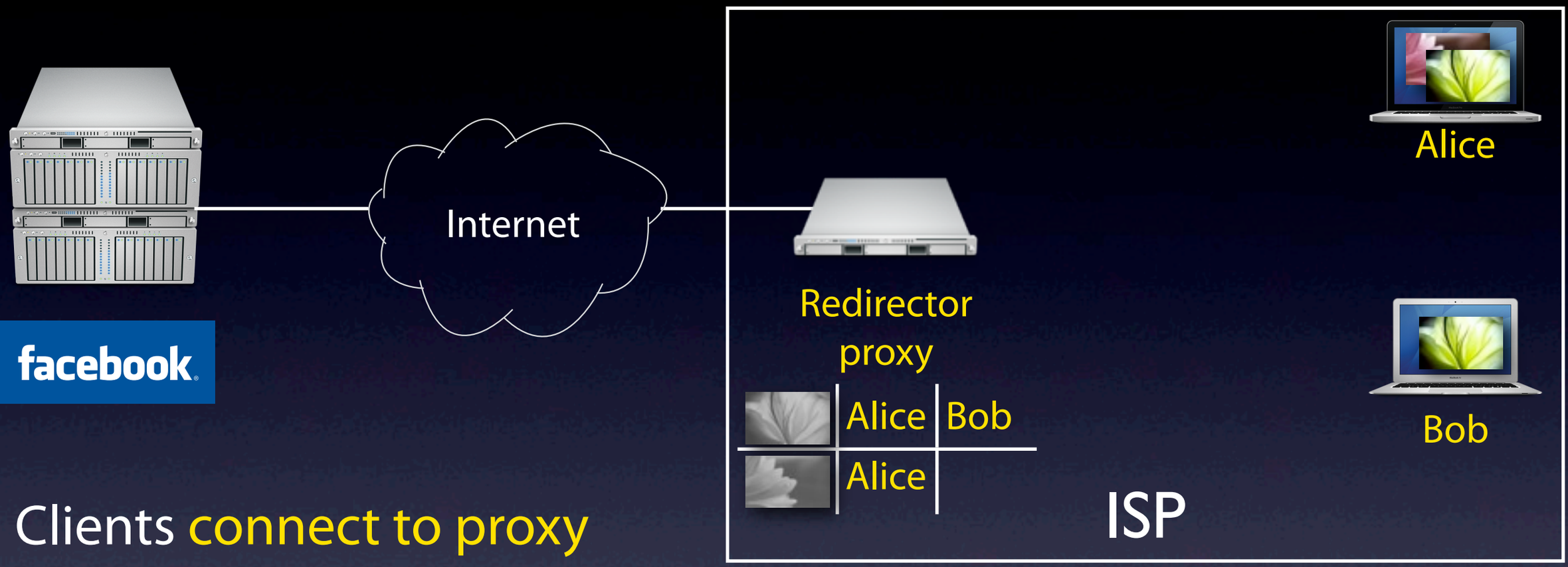
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



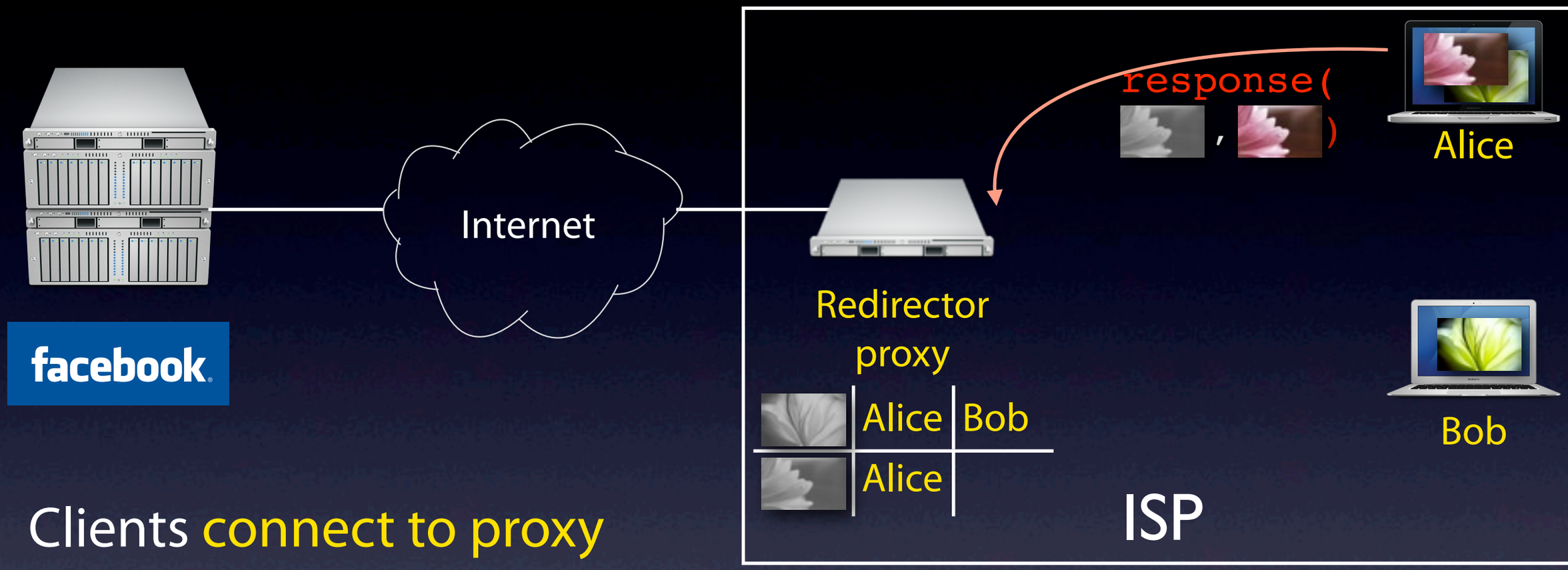
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



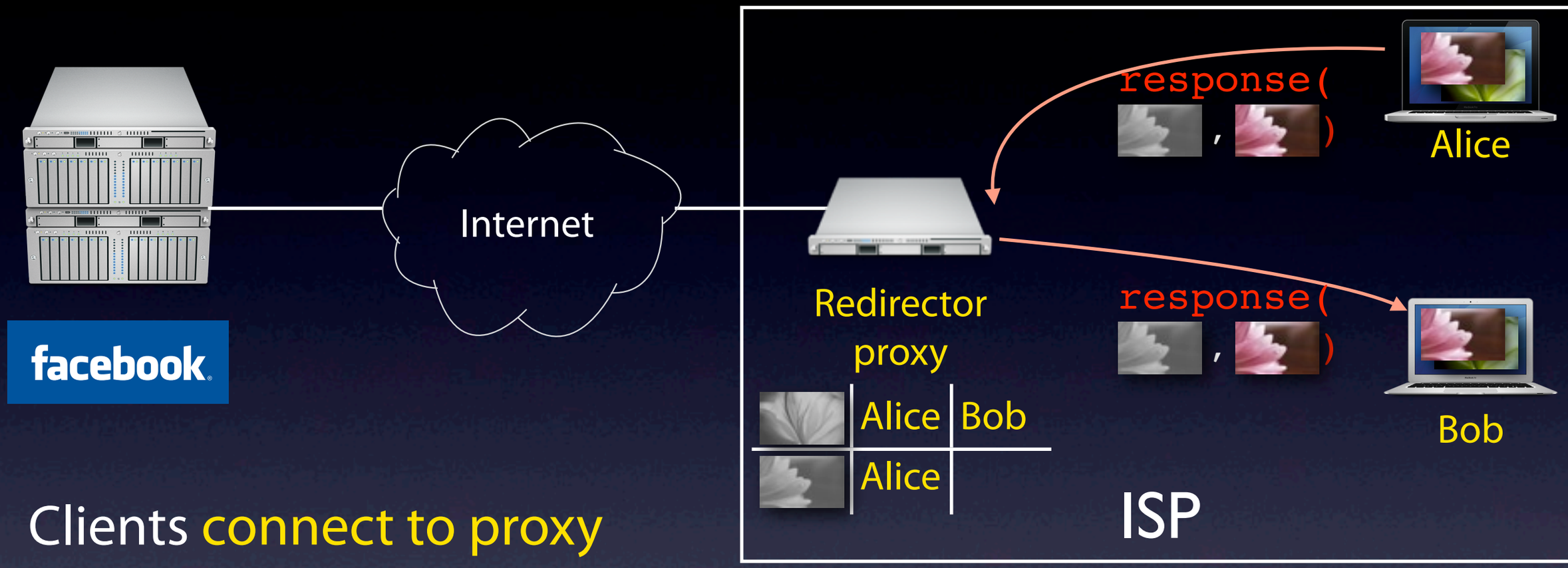
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



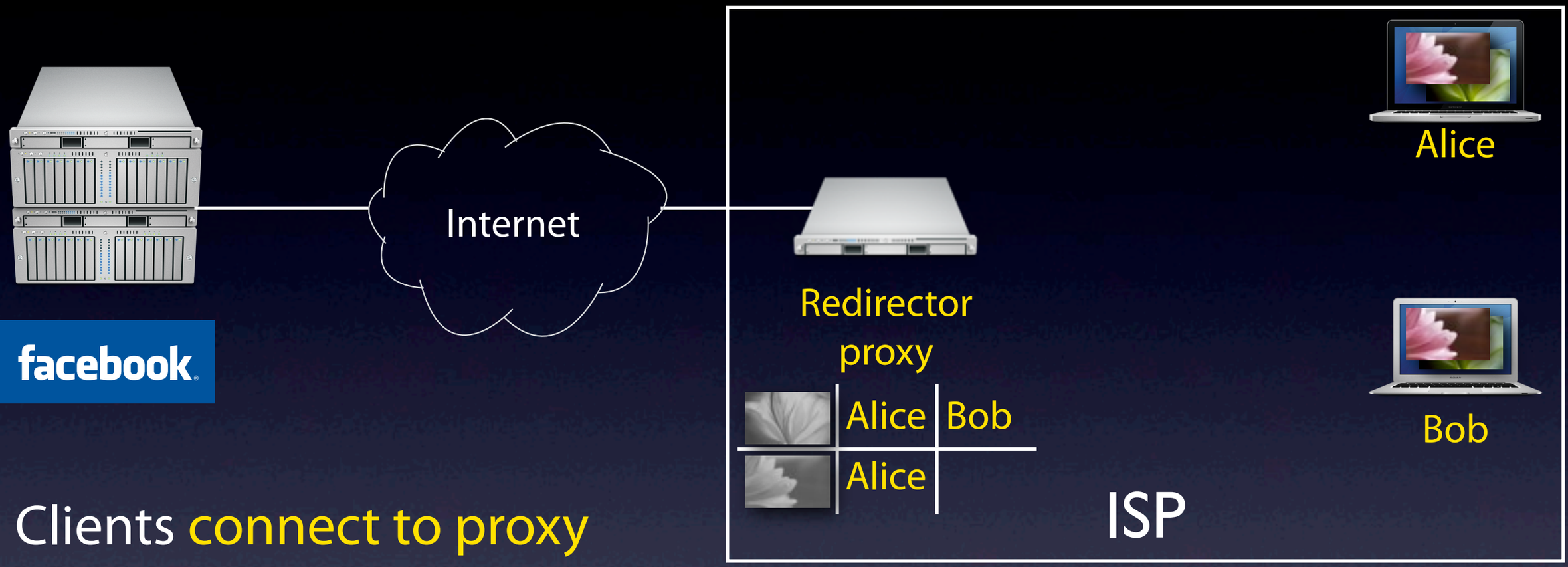
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



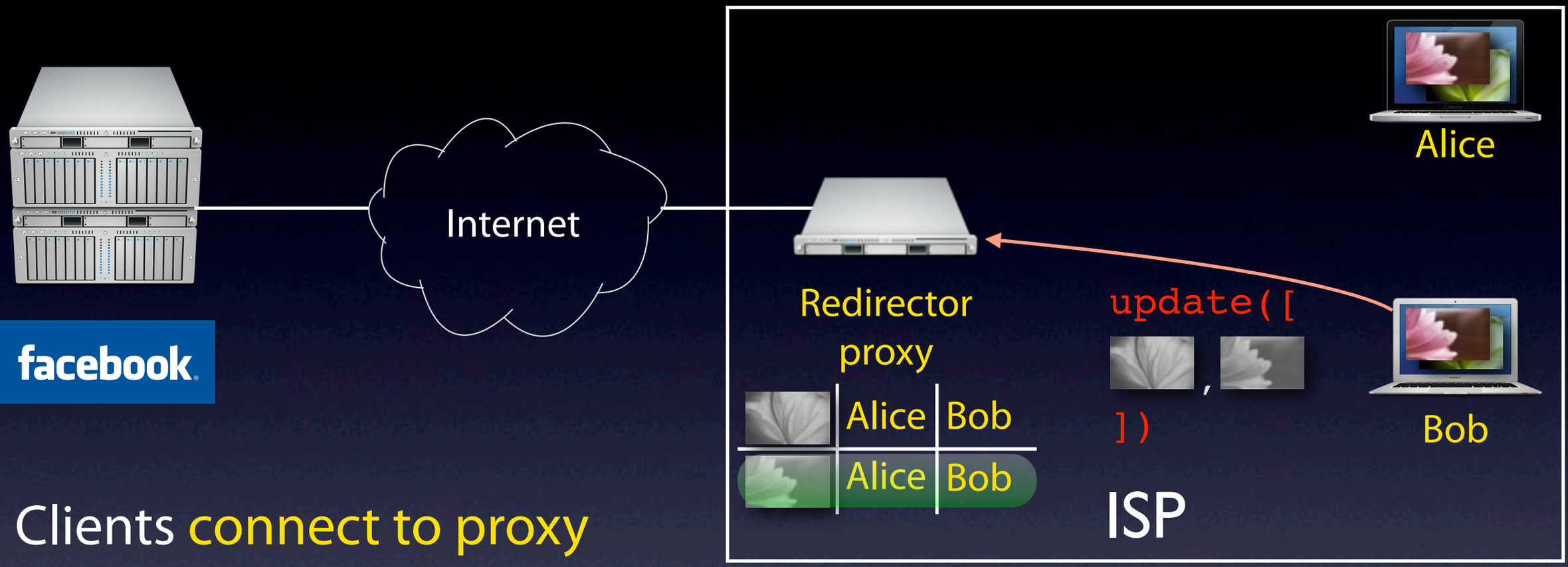
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



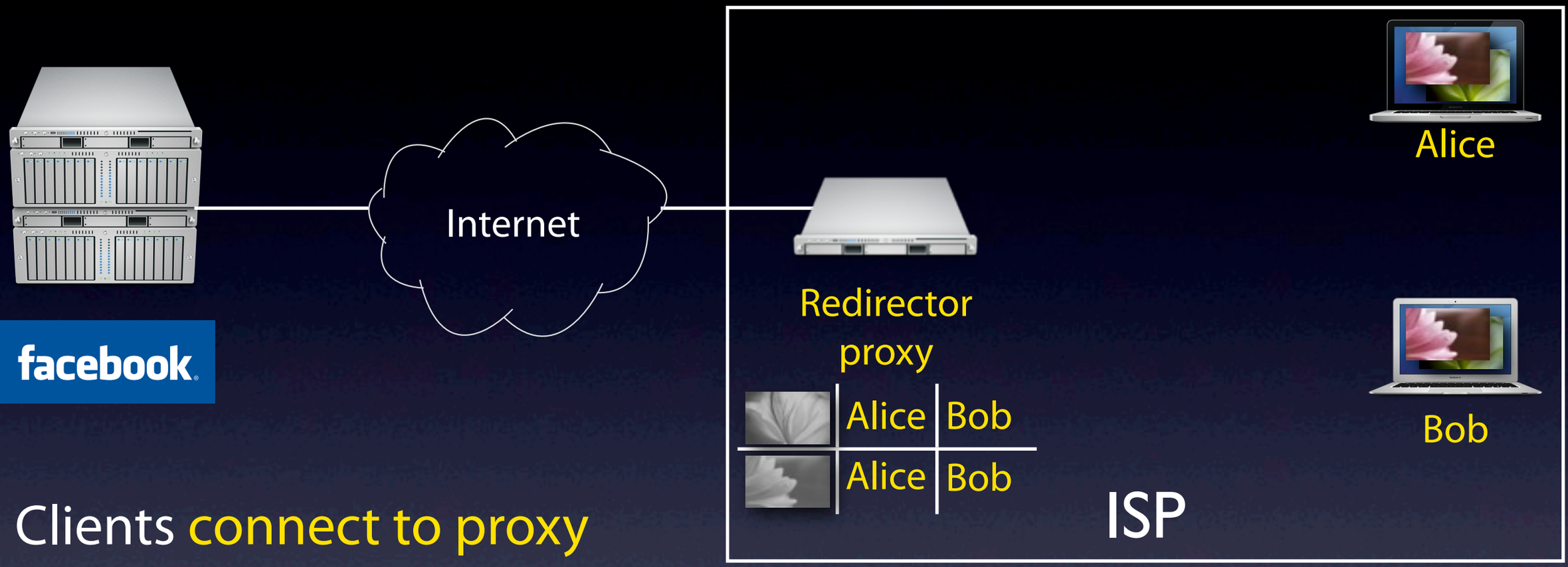
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



Clients connect to proxy

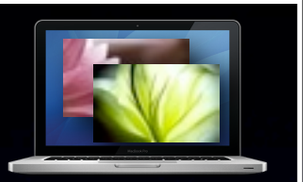
- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache

Redirector proxy



Alice

Bob

ISP

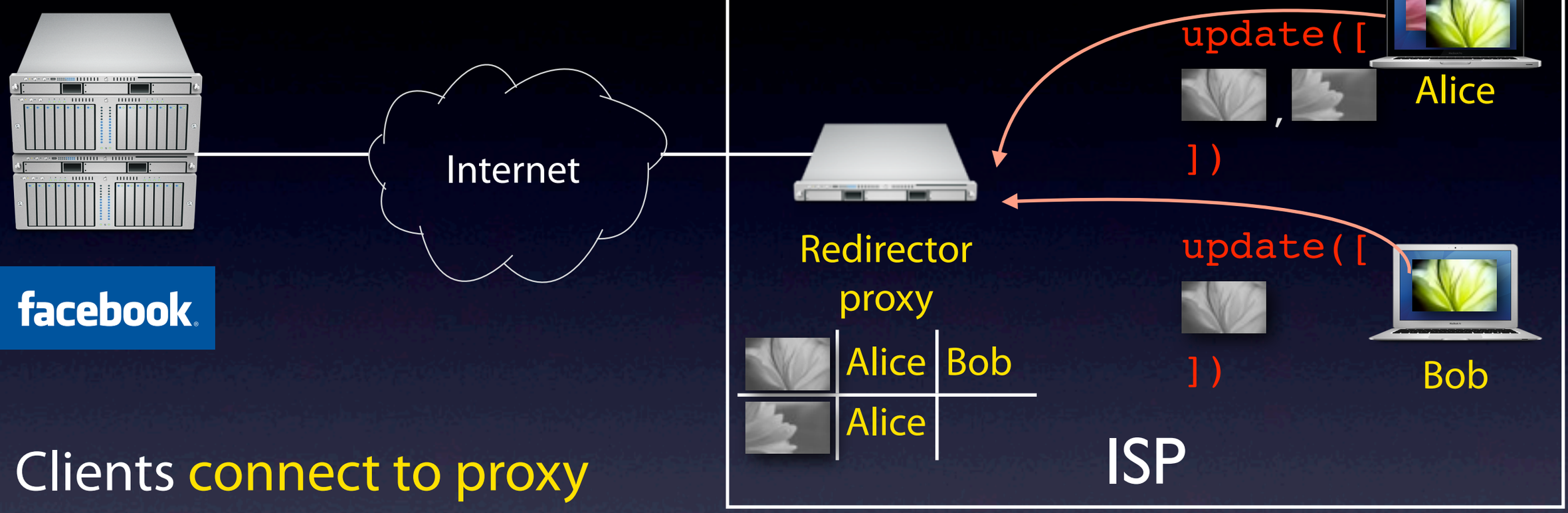
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



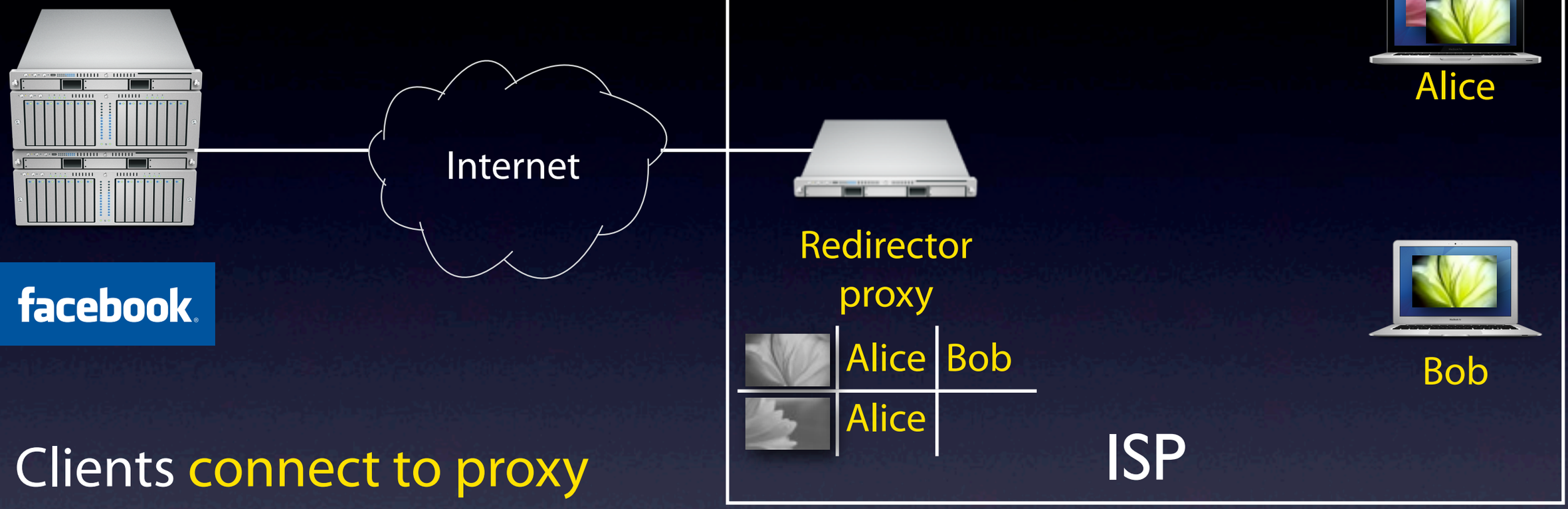
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



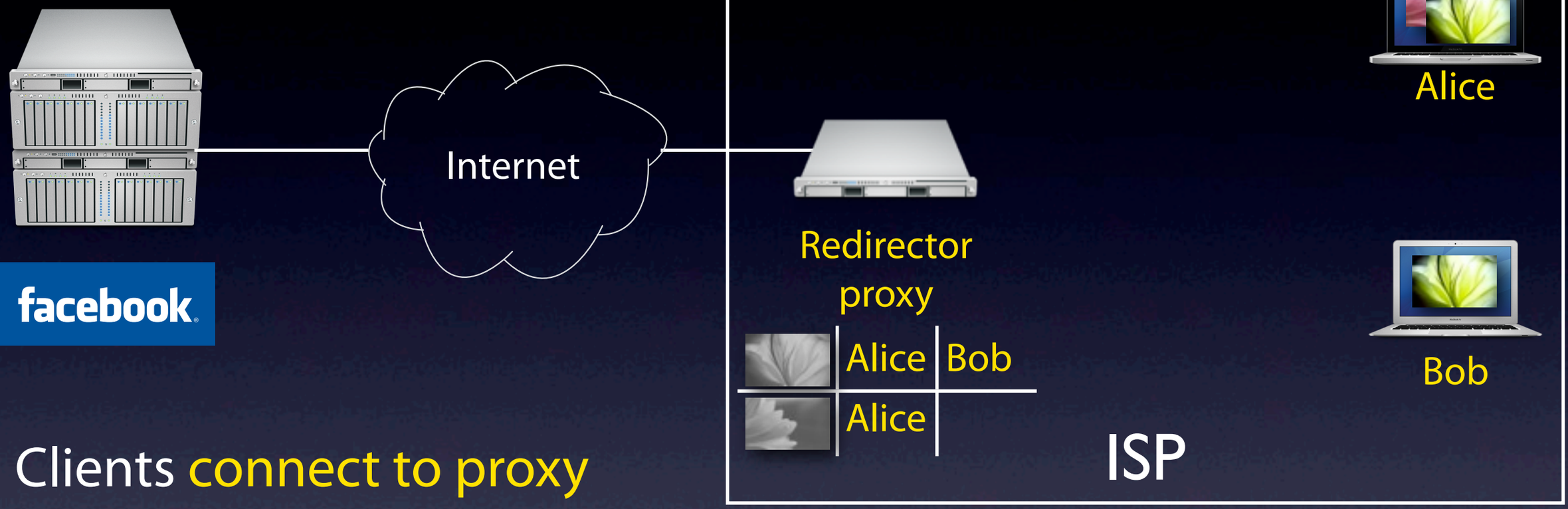
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



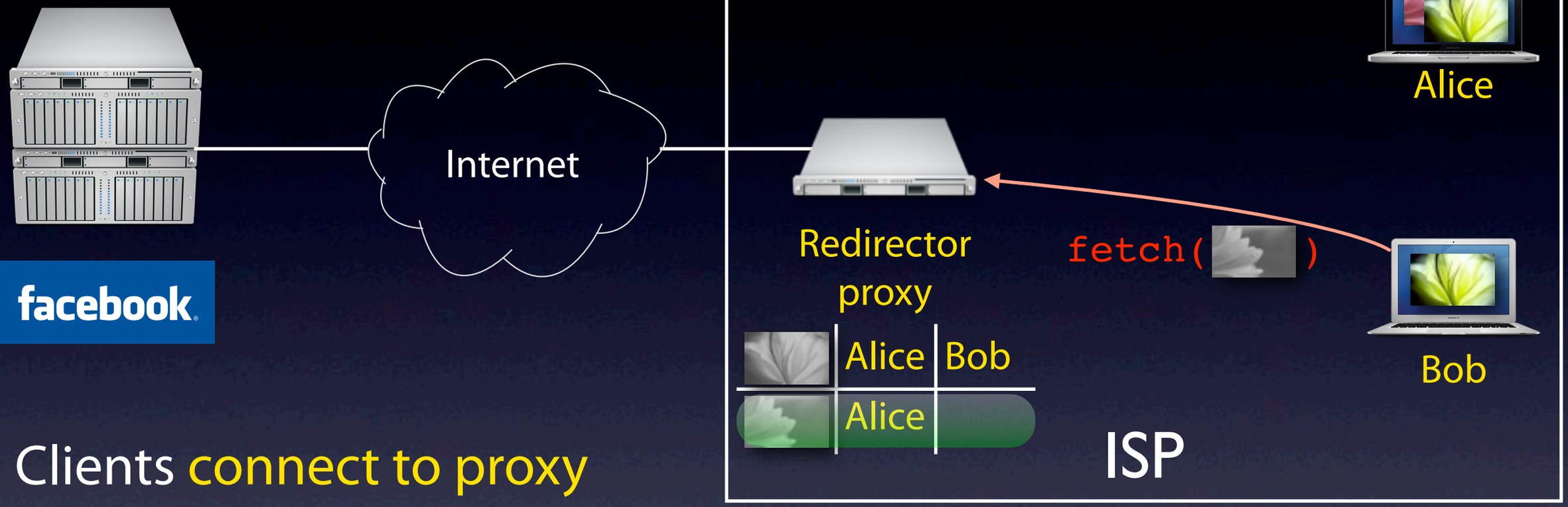
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



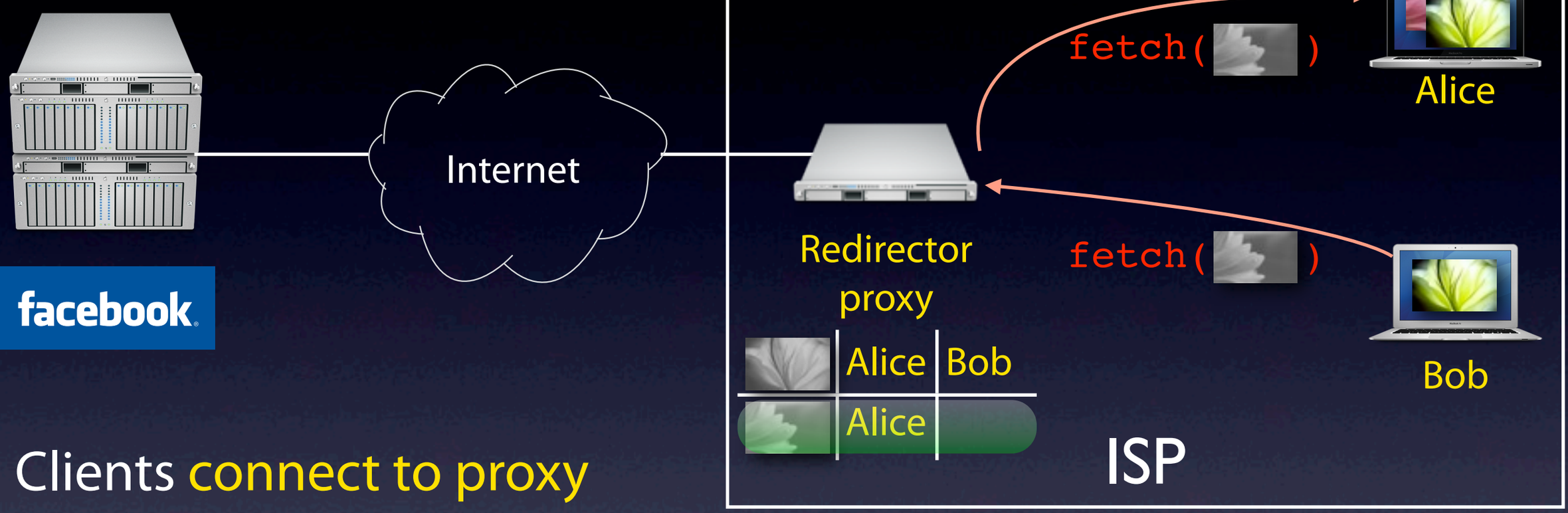
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



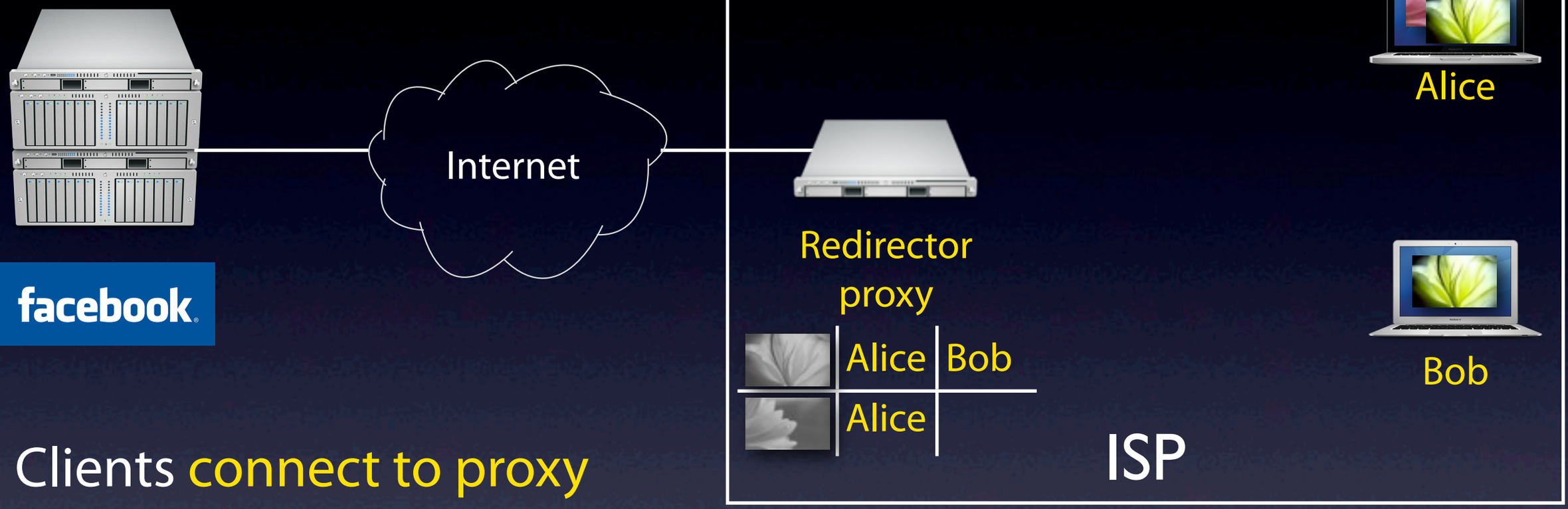
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



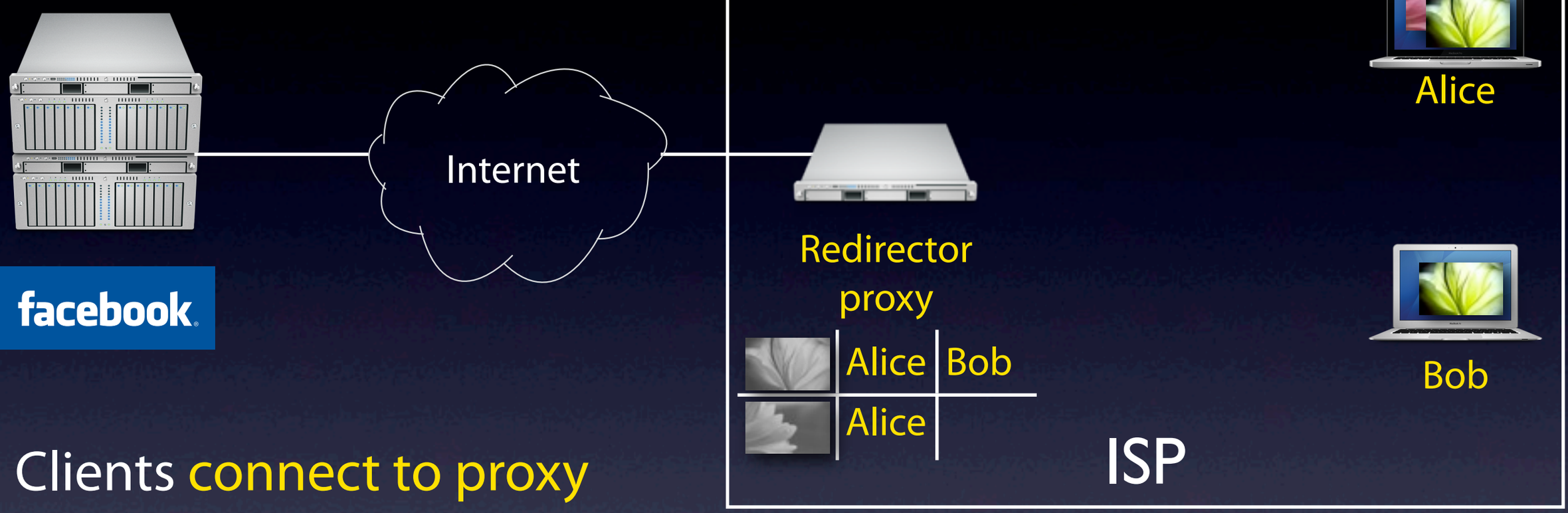
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



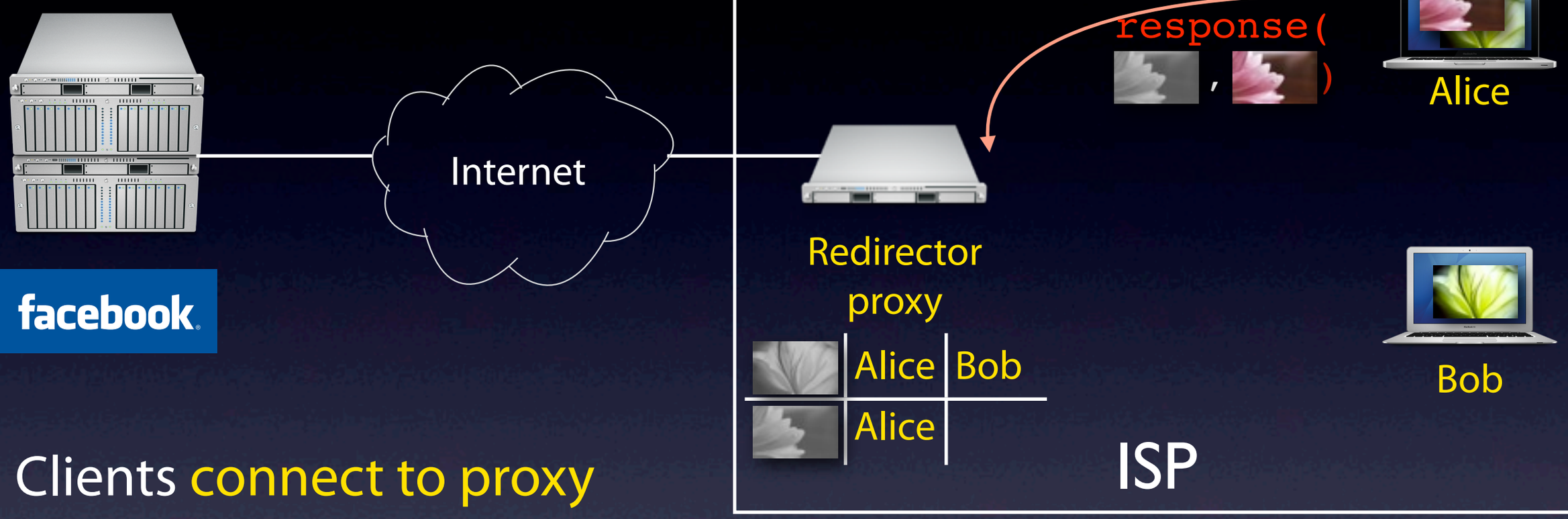
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



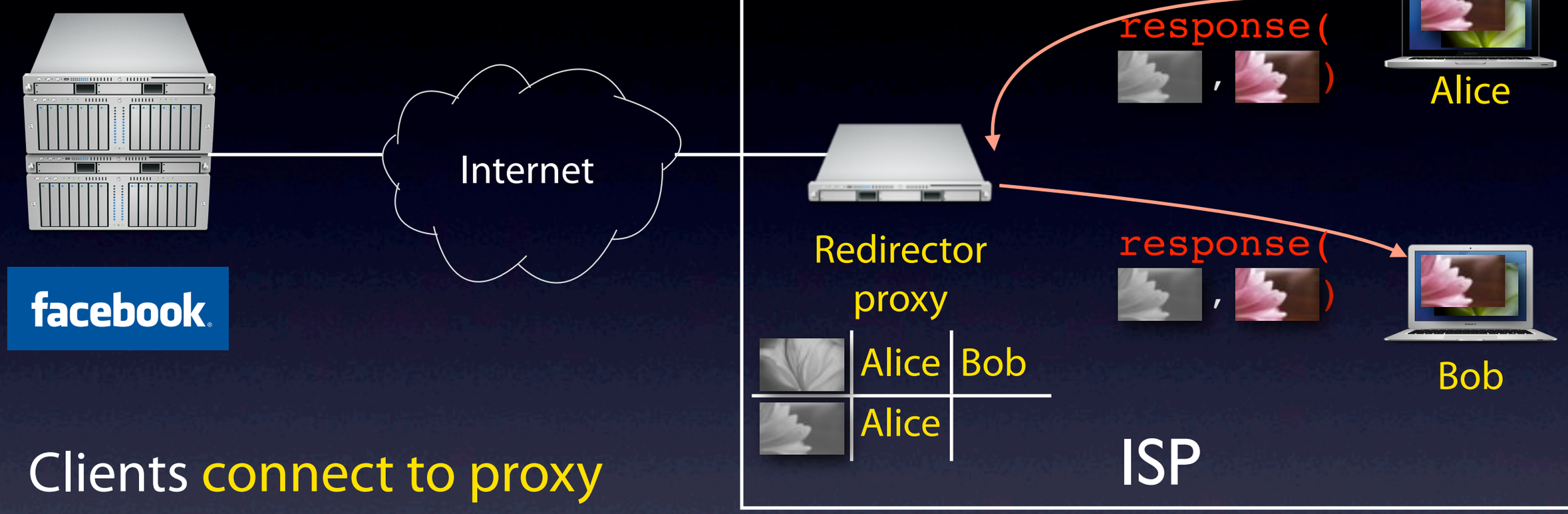
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



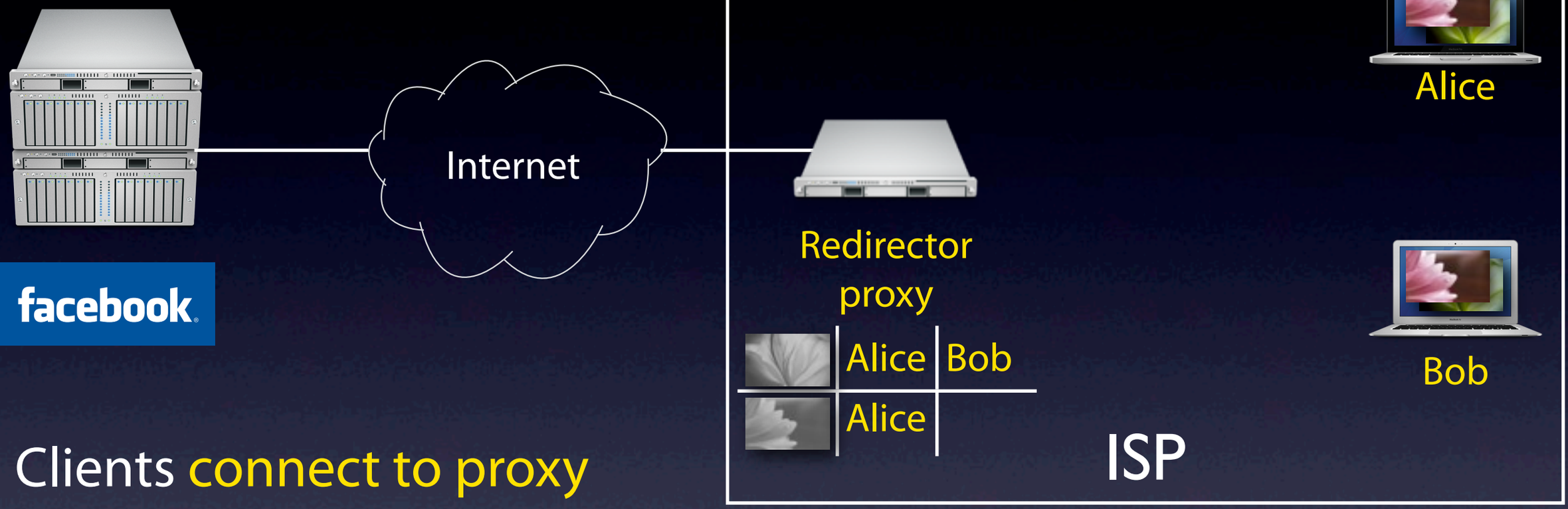
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



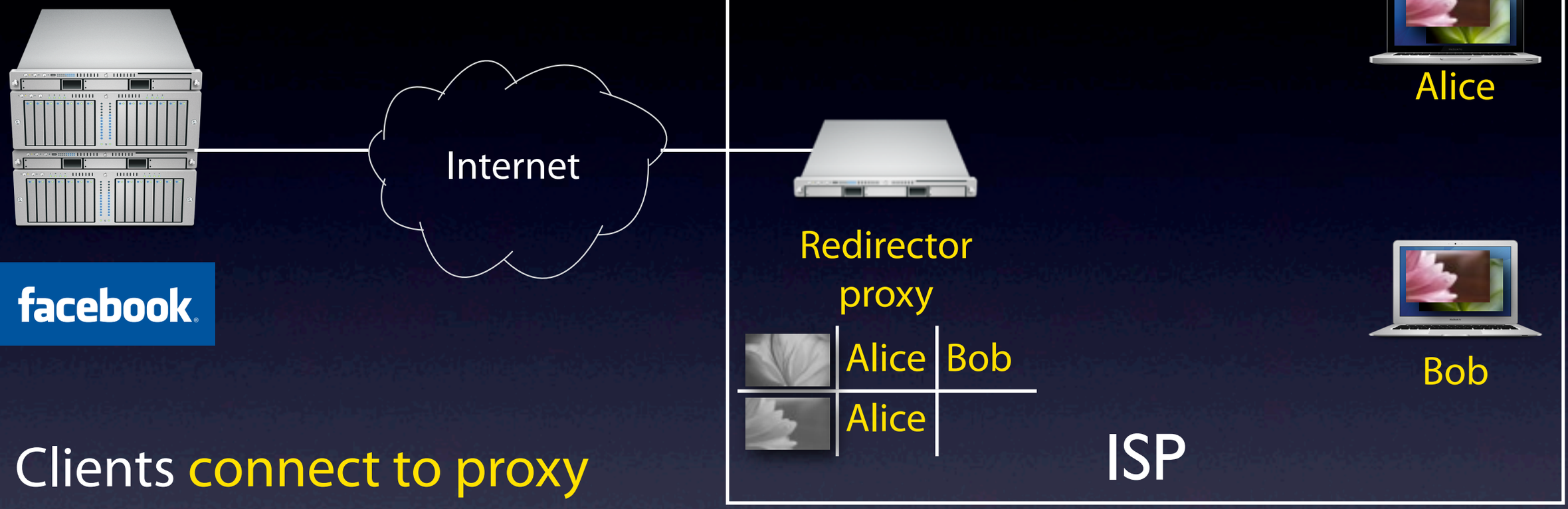
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



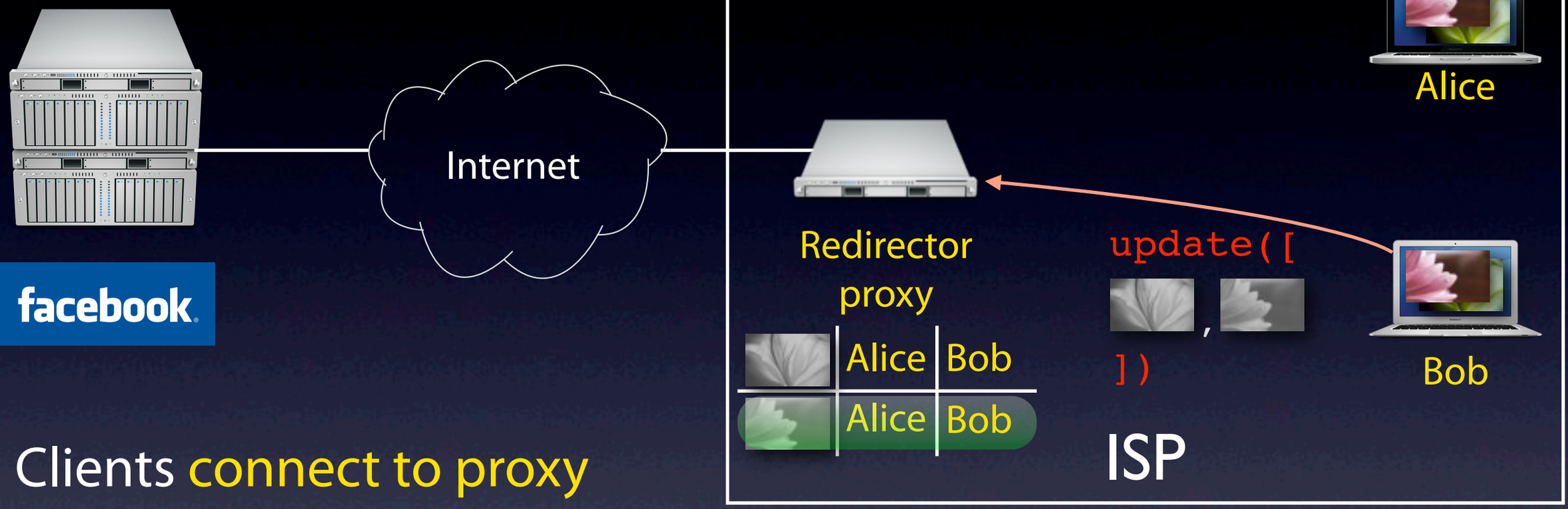
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



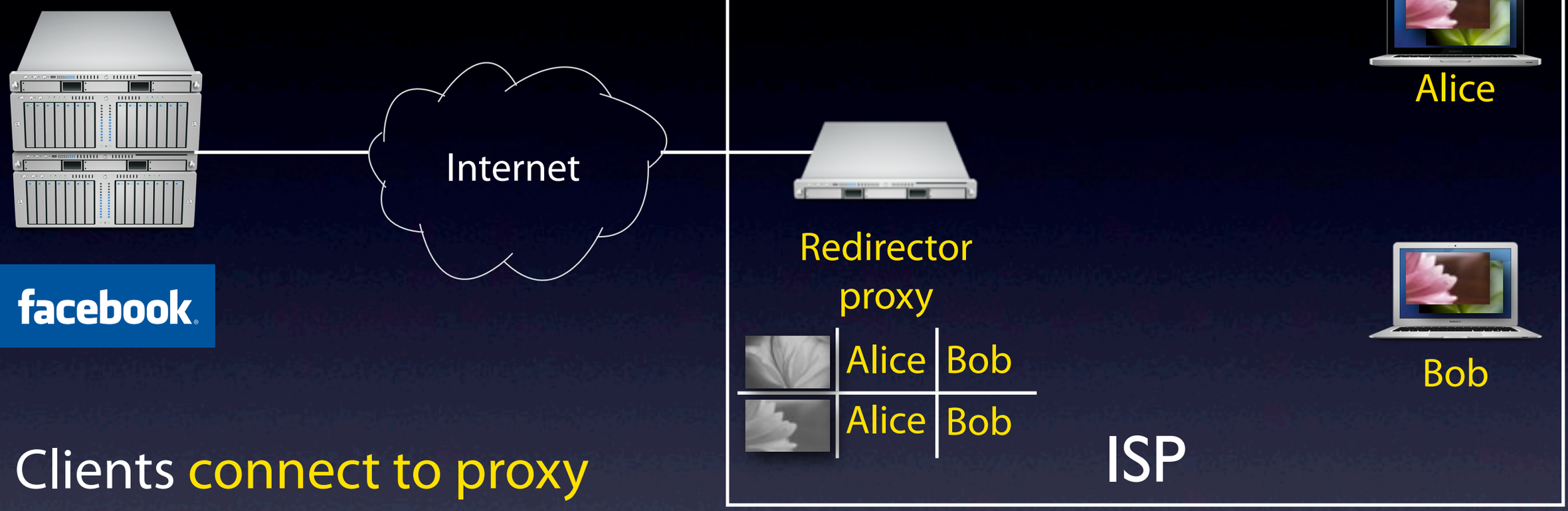
Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site

Distributed cache



Clients connect to proxy

- Inform proxy of locally stored content

Clients request content from proxy

- Proxy checks for other local clients
- Found: fetches content, forwards to requestor
- Not found: fetches content from origin site