Chaos Proxy

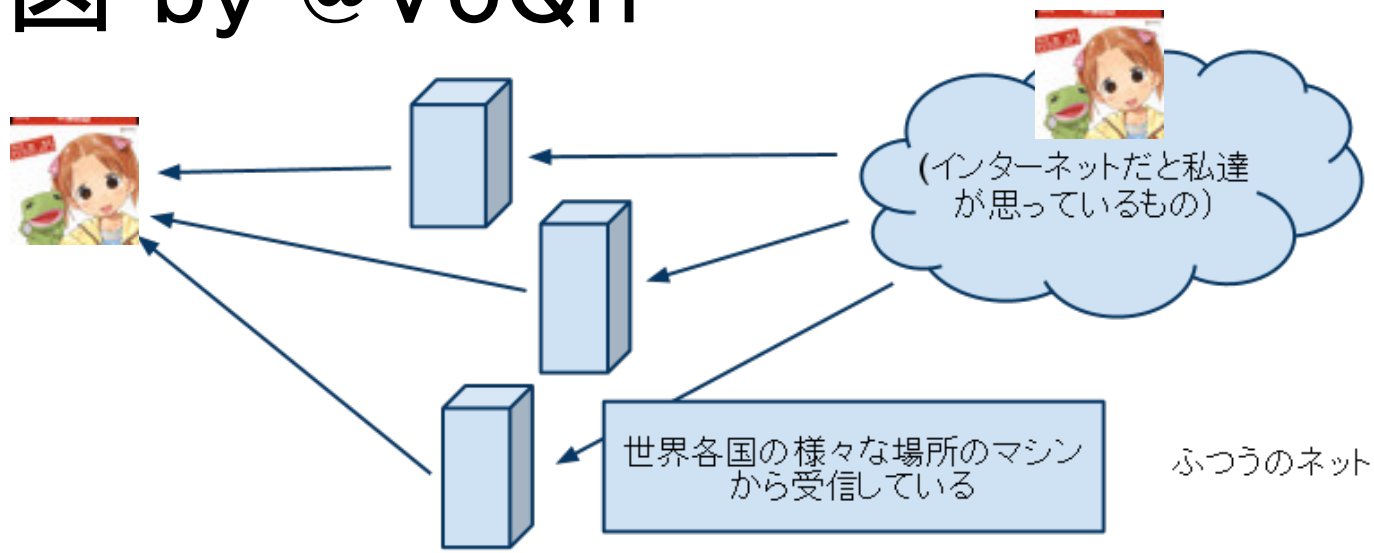
荒川智則

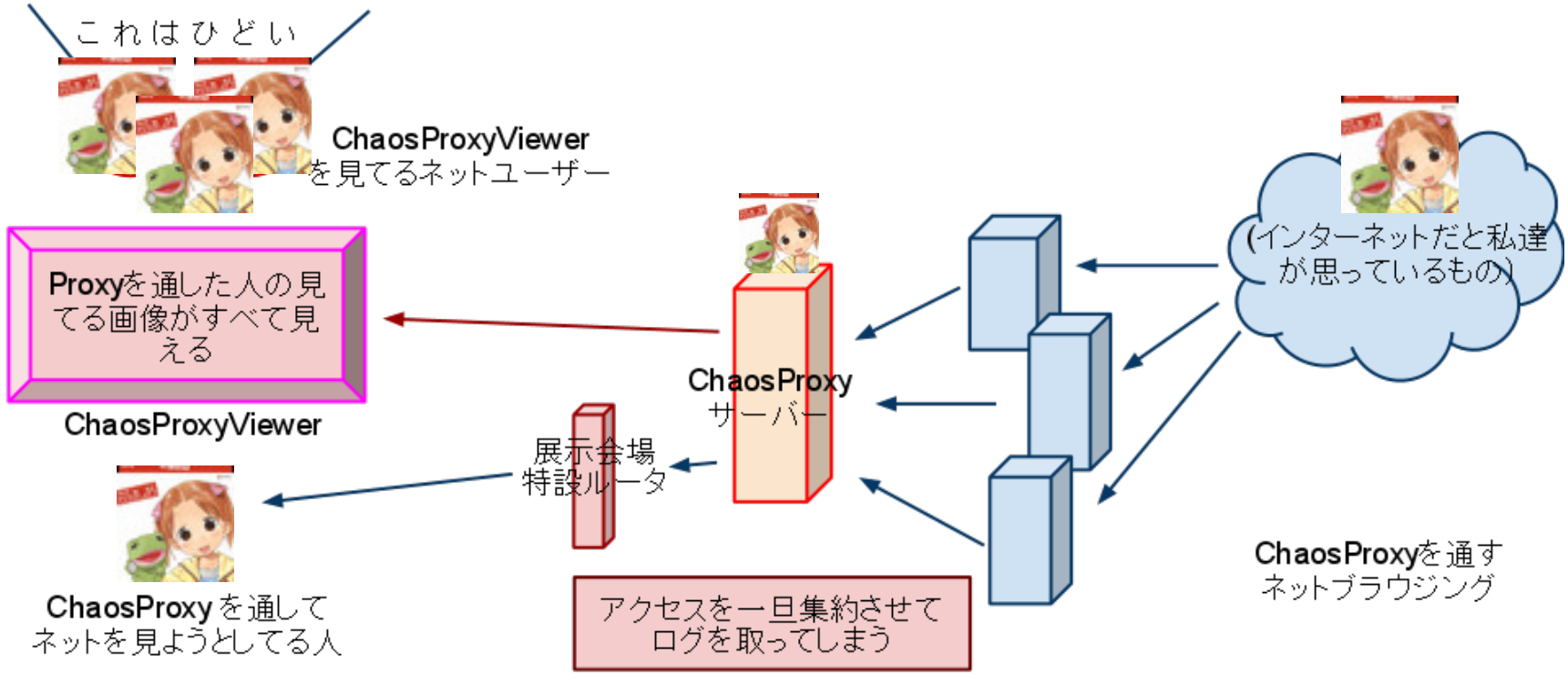
↓残りページ数

実現した

- 透過プロキシを使いWifiアクセスポイントをTwitterアカウントでログインするように強制
- 誰がどのアクセスポイントからどの画像にアクセスしたのか記録
- 上記画像一覧をWebサイト上でほぼリアルタイムにアニメーション表示
- プロキシへの詳細アクセスログをほぼリアルタイムにブラウザに出力
- ブラウザへ管理者の任意のおしらせメッセージを強制表示
- ブラウザを遠隔操作で強制リロードして最新のクライアントサイドコードを適用

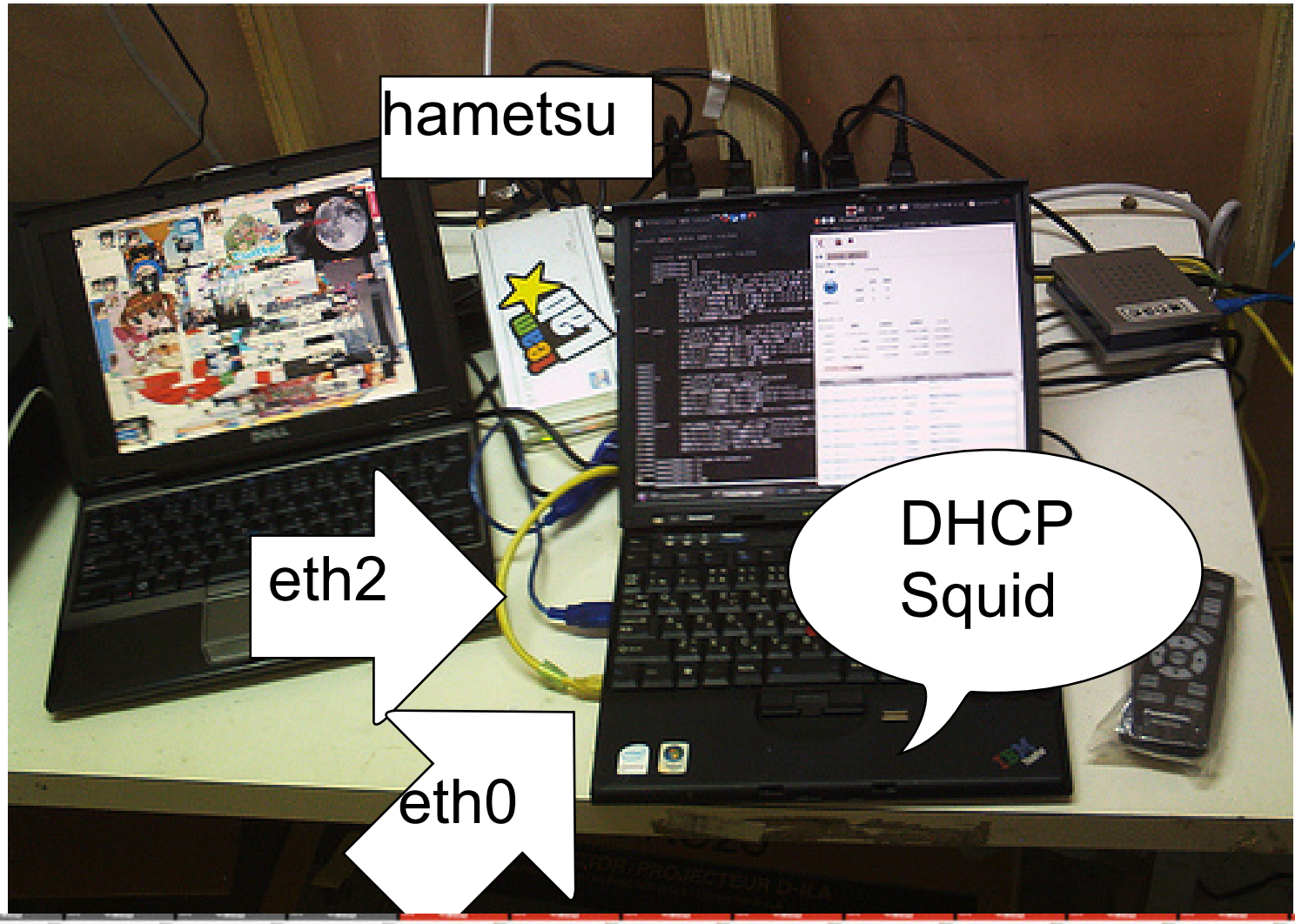
概要図 by @VoQn

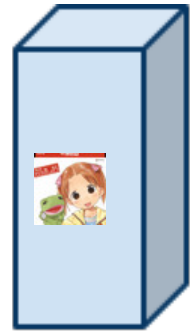

(インターネットだと私達
が思っているもの)
世界各国の様々な場所のマシン
から受信している
ふつうのネットブラウジング


これはひどい
ChaosProxyViewer
を見てるネットユーザー
Proxyを通した人の見
てる画像がすべて見
える
ChaosProxyViewer
ChaosProxy
サーバー
展示会場
特設ルータ
(インターネットだと私達
が思っているもの)
ChaosProxyを通して
ネットを見ようとしてる人
アクセスを一旦集約させて
ログを取ってしまう
ChaosProxyを通す
ネットブラウジング

会場ネットワークの構築 by @takano32

ifconfig eth2 192.168.32.1
iptables -t nat -A PREROUTING -i eth2 -p tcp -m tcp --dport 80 REDIRECT --to-ports 3128





ギークハウス浅草
便所鯖

Webrickは透過プロキシに対応してないからSquidを経由しないとダメダヨ♪

サーバー上のぷろせ

- /ttserv_tw_ctl
- /ttserv_img_ctl
- ruby rinda_server.rb
- ruby proxy.rb
- ruby websockets.rb
- ruby app.rb

proxy.rb < WebRick::HTTPProxyServer

- 端末ごとにユニークなPeer IDを生成
  - puid = "#{req.peeraddr[3]}:#{req.header['x-forwarded-for']}"
  - HTTPアクセスの度にpuidがtwitter名と紐付いているか確認、紐付いていなければtwitter.comに転送
- twitter.comにアクセスする度にアカウント情報の横取り
  - twitterのhomeはgzip圧縮されてるので解凍する
  - gzip解凍したあとの文字コードをbody.utf8する
  - あとはHpricotとかNokogiriとかでtwitter_name = 'span#me_name'、user_icon = 'img.side_thumb' あたりをぶっこ抜く

TokyoTyrant::RDBTBL

- テーブルデータベース
  - 認証用DBのスキーマ
    - {'puid' => puid.to_s, 'twitter_name' => twitter_name.to_s, 'user_icon' => user_icon.to_s, 'accessed_at' => Time.now.to_i.to_s }
  - 画像ログ用DBのスキーマ
    - {’uri’ => path.to_s, ’accessed_at’ => Time.now.to_i.to_s, ’puid’ =>puid.to_s}
- 主キーはレコード数＋１でauto increment
- Rubyバインディングで利用

app.rb < Sinatra::Application

- http://chaos.yuiseki.net/のルートはpublic/に配置したindex.html。動的に生成はしていない。
  - sinatraはTokyoTyrantのデータをjavascriptで利用するためのwrapperのような使い方
  - accessed_atはTT上でunixtime=数値型なのでソートしたり範囲指定して取得したりできる
- RDBTBLのqry.searchgetの結果は入れ子になったHash
  - require 'json/pure'してresult.to_jsonしてそのままレスポンスするだけ
  - クロスドメインのためレスポンスヘッダをいじる

http://chaos.yuiseki.net/の機能

- /
  - トップページ
- /users
  - 最近利用中のユーザー一覧
- /user/:screen_name
  - 各ユーザーの画像ログ一覧
- /onthe/:place
  - 各wifiアクセスポイントからアクセスしているユーザー一覧

rinda_server.rb

```
require 'drb/drb'
require 'rinda/tuplespace'
$ts = Rinda::TupleSpace.new
DRb.start_service('druby://:12345', $ts)
DRb.thread.join
```

- ↑コードはこれだけ
- 詳細な振る舞いはrindaクライアント側で記述する

websockets.rb < EventMachine::WebSocket

- proxy.rbはアクセスログをHashで
- スレッドでtuplespaceを監視してなにか変化があったらJSON.dumpしてEM::Channelにぶちこむ
- EM::Channelにつっこまれたら JSONはEventMachine::WebSocketが全クライアントにブロードキャストしてくれる

yわかったこと

- TTは各種パラメーターの調整が重要。パフォーマンス影響に直結する。。
- sinatraはシンプルで超柔軟なWebアプリフレームワーク。ちょっとしたトンチによってめちゃくちゃ使える感じになる。
- druby+rindaよくできてる。websocketsと相性良いかも。