DNS Summer Days 2017

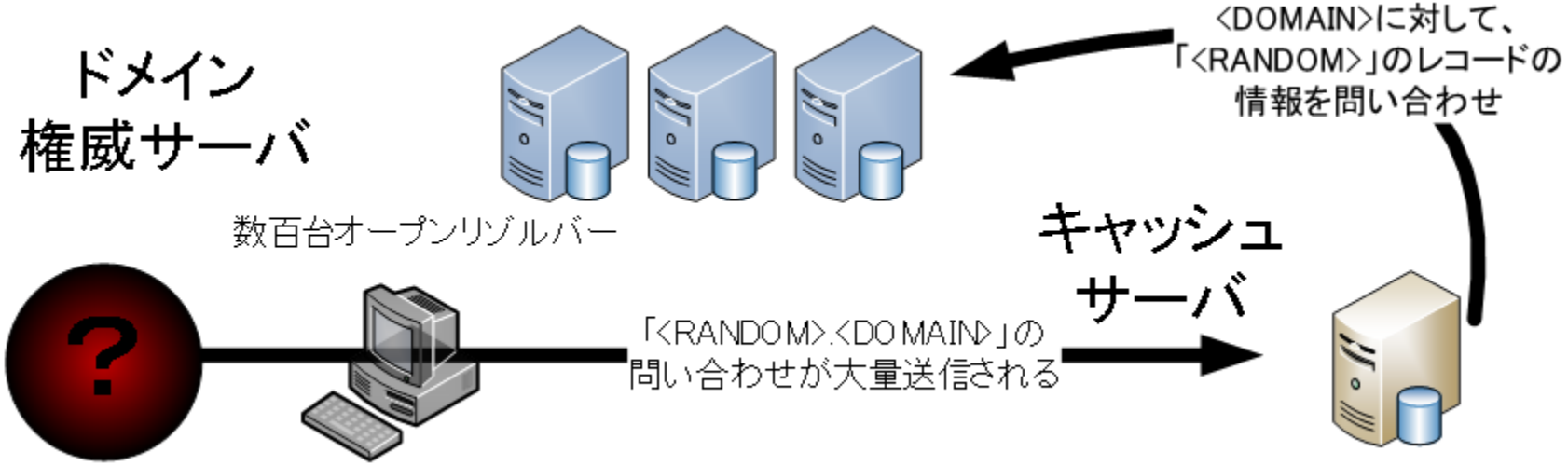
# UnboundでPRSD対策の実装

Asahi Net, Inc.
System Division

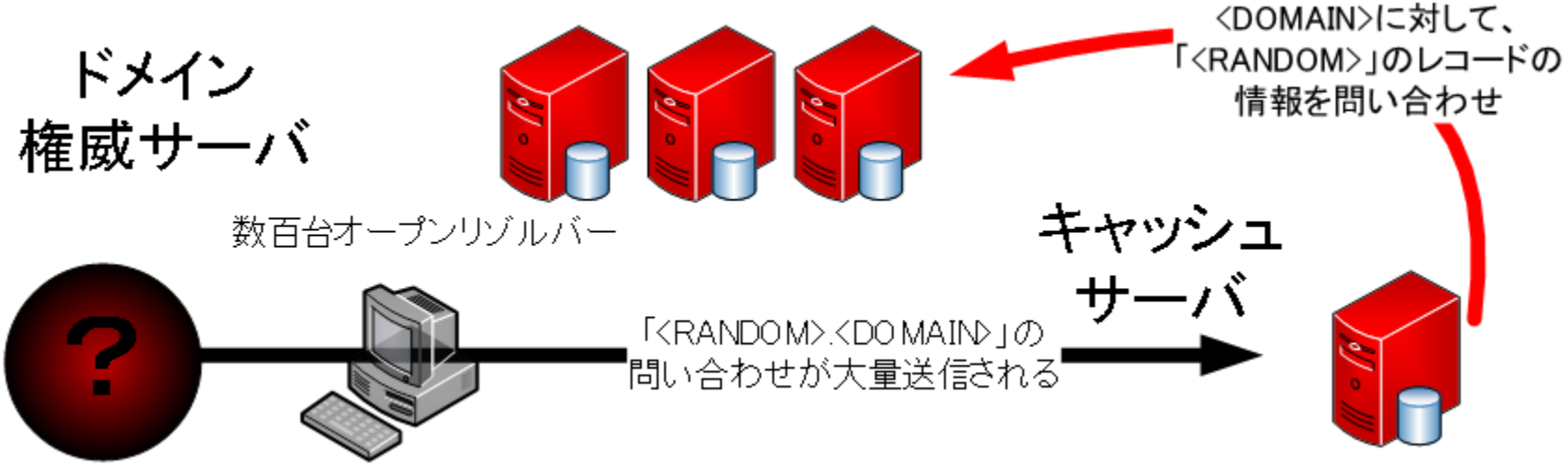
2017/6/19

つなぐをつくる、つなぐをささえる。
AsahiNet

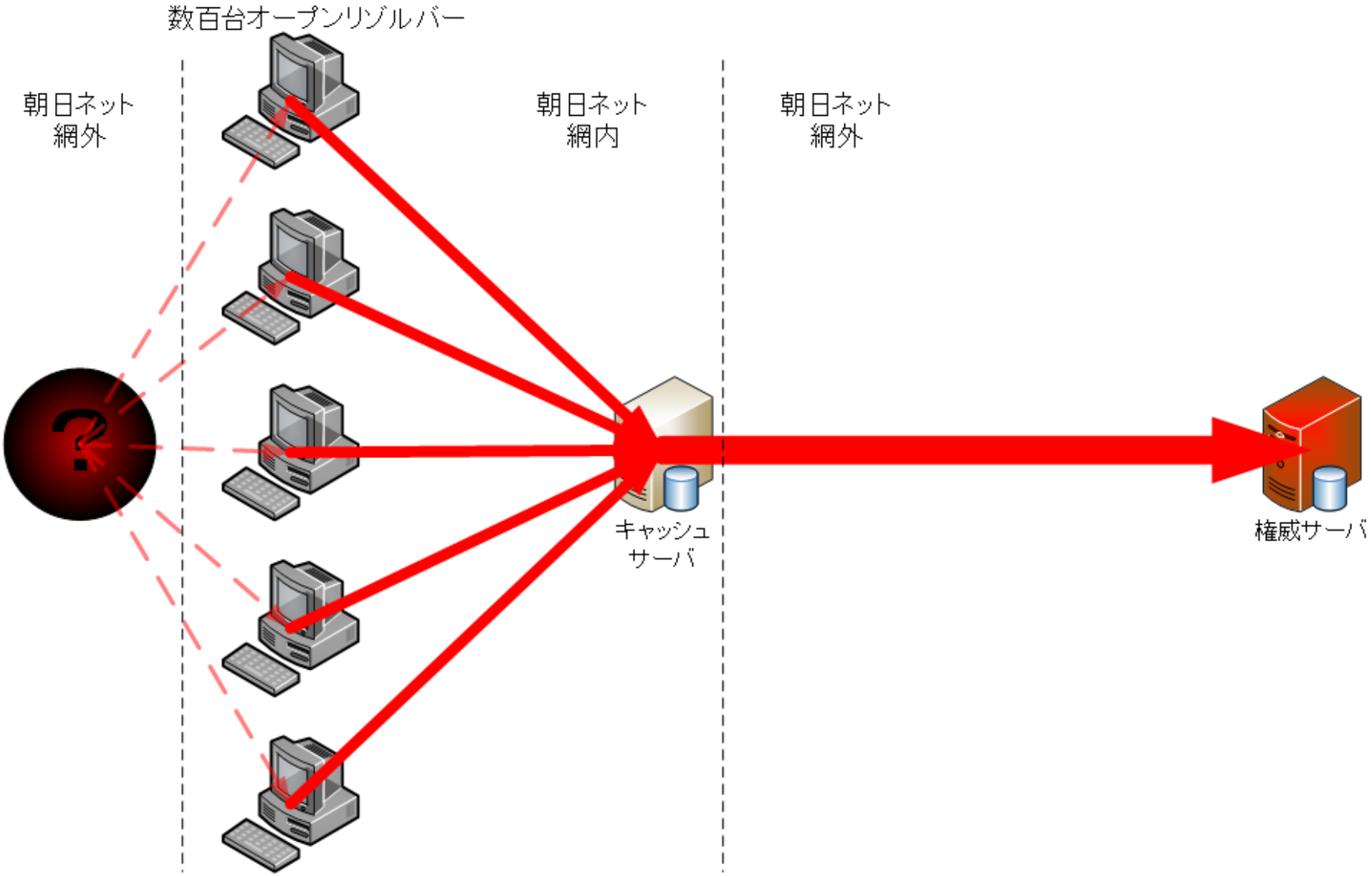
- キャッシュサーバーのメモリ消費・逼迫
- 帯域の逼迫
- 結果：
  - キャッシュサーバーのDDoS
  - 権威サーバーのDDoS



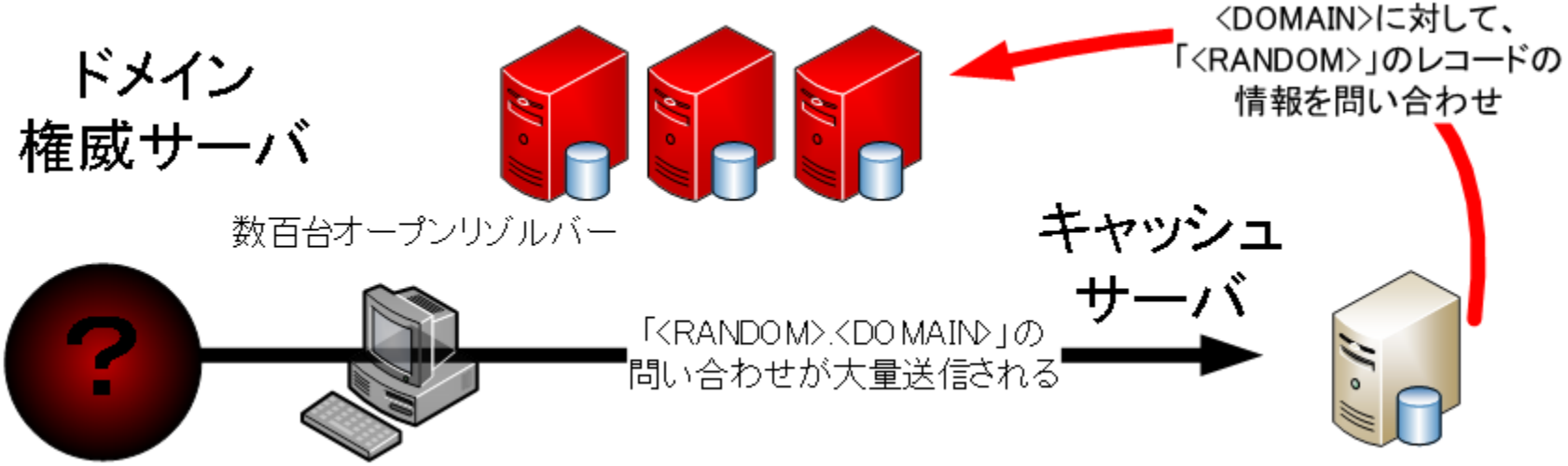
- キャッシュサーバーのメモリ消費・逼迫
- 帯域の逼迫
- 結果：
  - キャッシュサーバーのDDoS
  - 権威サーバーのDDoS



# PRSDクエリの本質：リソース消費



- BINDと違い、柔軟に大量の再起クエリに対応できている
  - 少なくとも自前の障害は避けられる
- Python APIで拡張可能



# Unboundに乗り換えるための課題（当時2014年）

- AAAAフィルター機能がない
  - 実装して、unboundのcontribに取り入れてもらった
- 攻撃中、BINDより、帯域を使ってしまう
  - やはり、根本対策が必要

# 思いついた「迎撃」対策：攻撃対象のドメインの識別

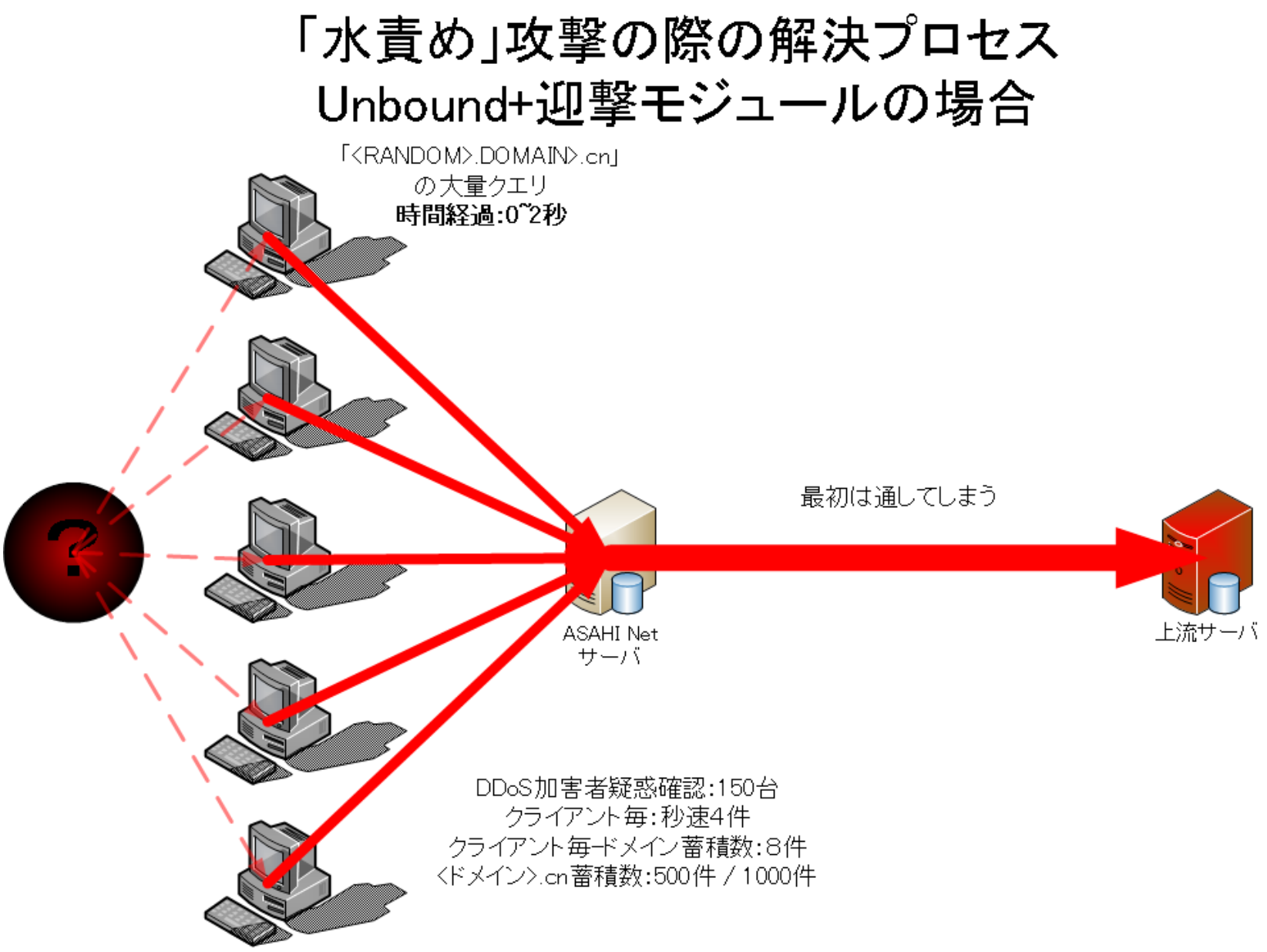
- ランダムサブドメインと言っても、攻撃の的が権威サーバーなら、委任ポイント情報をもとに特定できるはず
- unbound-control等で、infra cacheから引き出せる
- Python APIを拡張して、取得できる改造をした
- Unbound本家で導入済み

```
atson1@eagle04:~$ sudo unbound-control lookup .
[...]
Delegation with 13 names, of which 12 can be examined to query further addresses.
It provides 13 IP addresses.
199.7.91.13             rto 715 msec, ttl 395, ping 127 var 147 rtt 715, tA 0, tAAAA 0, tother 0, EDNS 0 probed.
192.33.4.12             rto 862 msec, ttl 33, ping 238 var 156 rtt 862, tA 0, tAAAA 0, tother 0, EDNS 0 probed.
193.0.14.129            rto 442 msec, ttl 705, ping 18 var 106 rtt 442, tA 0, tAAAA 0, tother 0, EDNS 0 probed.
198.41.0.4              rto 834 msec, ttl 591, ping 90 var 186 rtt 834, tA 0, tAAAA 0, tother 0, EDNS 0 probed.
192.203.230.10          rto 466 msec, ttl 805, ping 58 var 102 rtt 466, tA 0, tAAAA 0, tother 0, EDNS 0 probed.
192.58.128.30           rto 668 msec, ttl 21, ping 132 var 134 rtt 668, tA 0, tAAAA 0, tother 0, EDNS 0 probed.
192.36.148.17           rto 298 msec, ttl 239, ping 54 var 61 rtt 298, tA 0, tAAAA 0, tother 0, EDNS 0 probed.
192.228.79.201          rto 462 msec, ttl 4, ping 162 var 75 rtt 462, tA 0, tAAAA 0, tother 0, EDNS 0 probed.
192.5.5.241             rto 788 msec, ttl 394, ping 72 var 179 rtt 788, tA 0, tAAAA 0, tother 0, EDNS 0 probed.
128.63.2.53             rto 1440 msec, ttl 251, ping 148 var 323 rtt 1440, tA 0, tAAAA 0, tother 0, EDNS 0 probed.
192.112.36.4            rto 703 msec, ttl 464, ping 99 var 151 rtt 703, tA 0, tAAAA 0, tother 0, EDNS 0 probed.
199.7.83.42             rto 1020 msec, ttl 33, ping 160 var 215 rtt 1020, tA 0, tAAAA 0, tother 0, EDNS 0 probed.
202.12.27.33            rto 522 msec, ttl 837, ping 26 var 124 rtt 522, tA 0, tAAAA 0, tother 0, EDNS 0 probed.
```

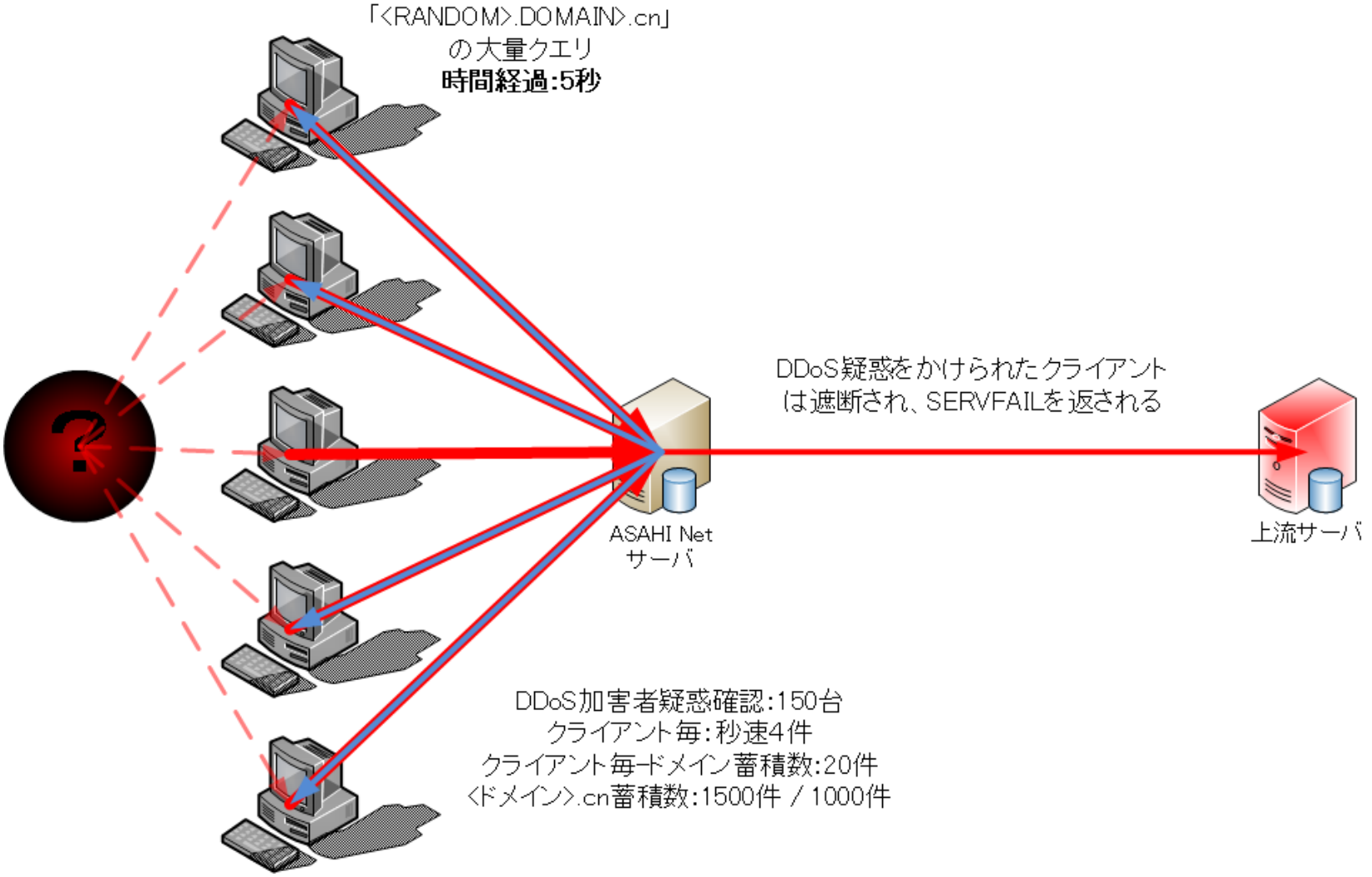
# 思いついた「迎撃」対策：専用モジュール「geigeki」の開発

- 「ドメイン」、「クライアント」、「クライアント→ドメイン」ごとに、下記の情報を集計する：
  - NXDOMAINの数
  - 返ってきたRRの数
  - クエリの頻度
- 上記数値を持って、ドメインやクエリの意味を想定する（例：NXDOMAIN 99%で、RRが一つもなければ、意味ない）
- 統計は、5分間蓄積するシステム
- 下記条件で「クライアント」から「ドメイン」に対するクエリをSERVFAILさせる：
  - ドメインが「攻撃されている」と判断される量を超えた
  - クライアントがそのドメインを刺激するクエリを多少出している（秒間5件等）

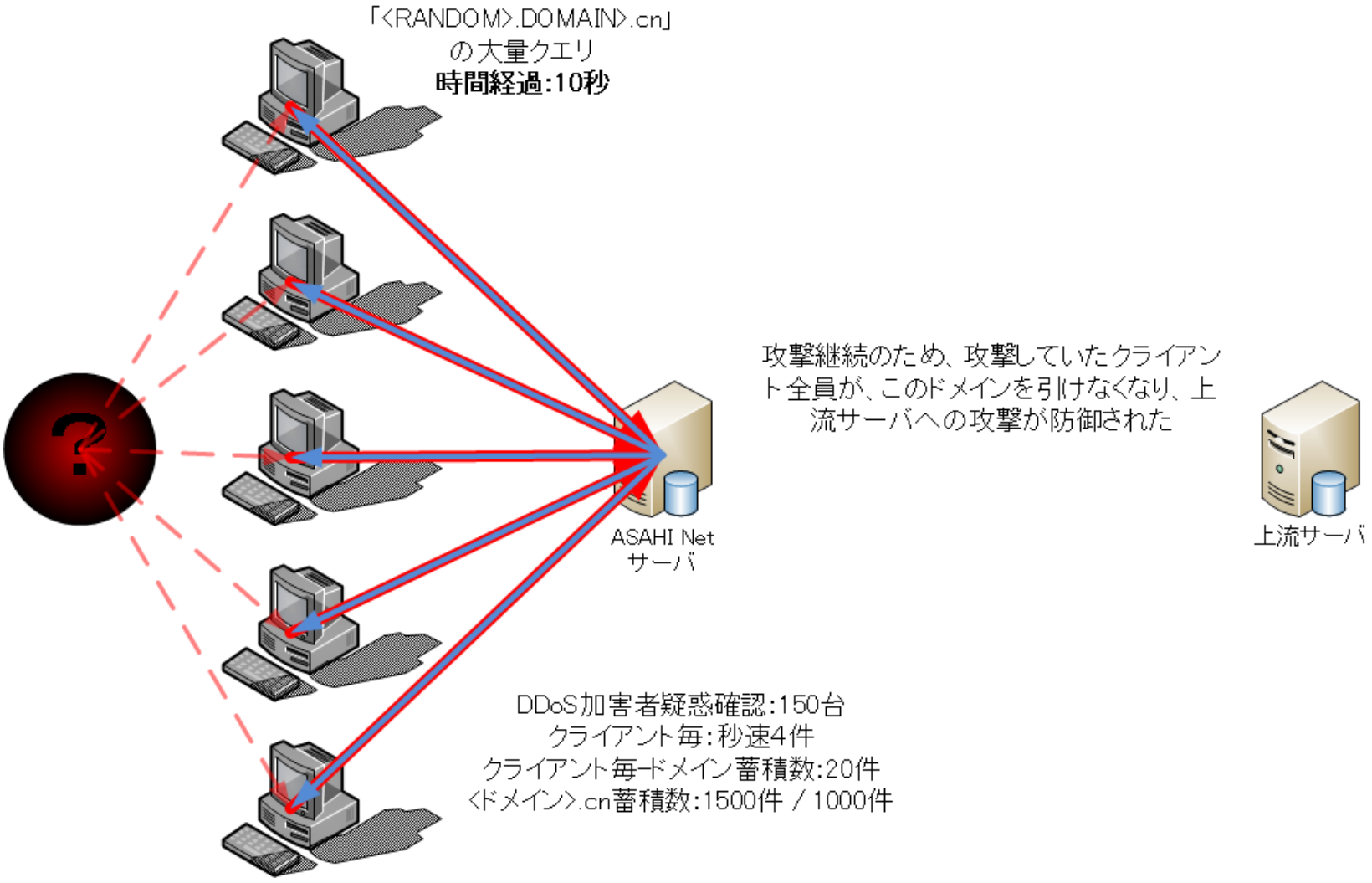
# 思いついた「迎撃」対策：専用モジュール「geigeki」の効果



# 「水責め」攻撃の際の解決プロセス
# Unbound+迎撃モジュールの場合



# 「水責め」攻撃の際の解決プロセス
# Unbound+迎撃モジュールの場合



# 迎撃対策のメリット

- 攻撃の的になってる、特定のドメインだけがブロックされる
- 他のクエリが通常どおり、利用者への迷惑を避ける
- DNSサーバーでSERVFAILを返すので、上流（最寄ピアや権威サーバー）への多量のクエリを避ける
- 無駄なものを極力キャッシュしないので、メモリ逼迫も避ける
- 解析はリアルタイムでメモリ内で行うので、遅延なしに対応する

# 迎撃対策の注意事項（1）

- 現在の実装はさらに最適化可能なはず
- 大手業者（某検索エンジン、某メーカー等）のホワイトリストが必要
- PTRレコード：
  - 上記と同様、ホワイトリスト化する必要がある
- IPv6クエリ：
  - とりあえず、クライアントを「/64」の単位でまとめる

# 迎撃対策の注意事項(2)

- 現在の実装では、対象は「ドメイン名」であって、「権威サーバー」ではない(ロジックの応用は可能)
- 現在、細かいしきい値は、観測の上、調整が必要
  - もちろん、「実際ブロックしない」デバグモードがある
  - 仮運用と調整の上、本番化する
- 再起動したら、メモリ内の情報が消失する
  - だが寿命が短い情報しかないので、さほど問題ない

# ご質問