「根元」は攻略されたのか～

DNSキャッシュポイズニング攻撃とその対策について改めて考える

～

藤原 和典
株式会社日本レジストリサービス/JPRS
fujiwara@jprs.co.jp

2014/9/12

信学技報の正誤表

- 4.4.1フルリゾルバ運用者の対策
  - harden-refferal-path → harden-referral-path
- 表2
  - 2^16 → 2^{16}
- 6. 現実の攻撃と対策
  - 健在化　→ 顕在化

Copyright © 2014 Japan Registry Services Co., Ltd.

根元？

- ルートDNSサーバの構造的な問題？
- プロトコル？
- TLDではないはず？

Copyright © 2014 Japan Registry Services Co., Ltd.

概要

- DNSへの攻撃の紹介と対策
- MüllerのNode re-delegation attack
  - 2008年に公開されたホワイトペーパー
  - 上記に加え、以下の内容を紹介
    - 注入可能なドメイン名
    - 攻撃にかかる時間の期待値
    - 攻撃ツール試作、攻撃実験、期待値との比較
    - 対策
    - ソースポートランダマイゼーションの普及度

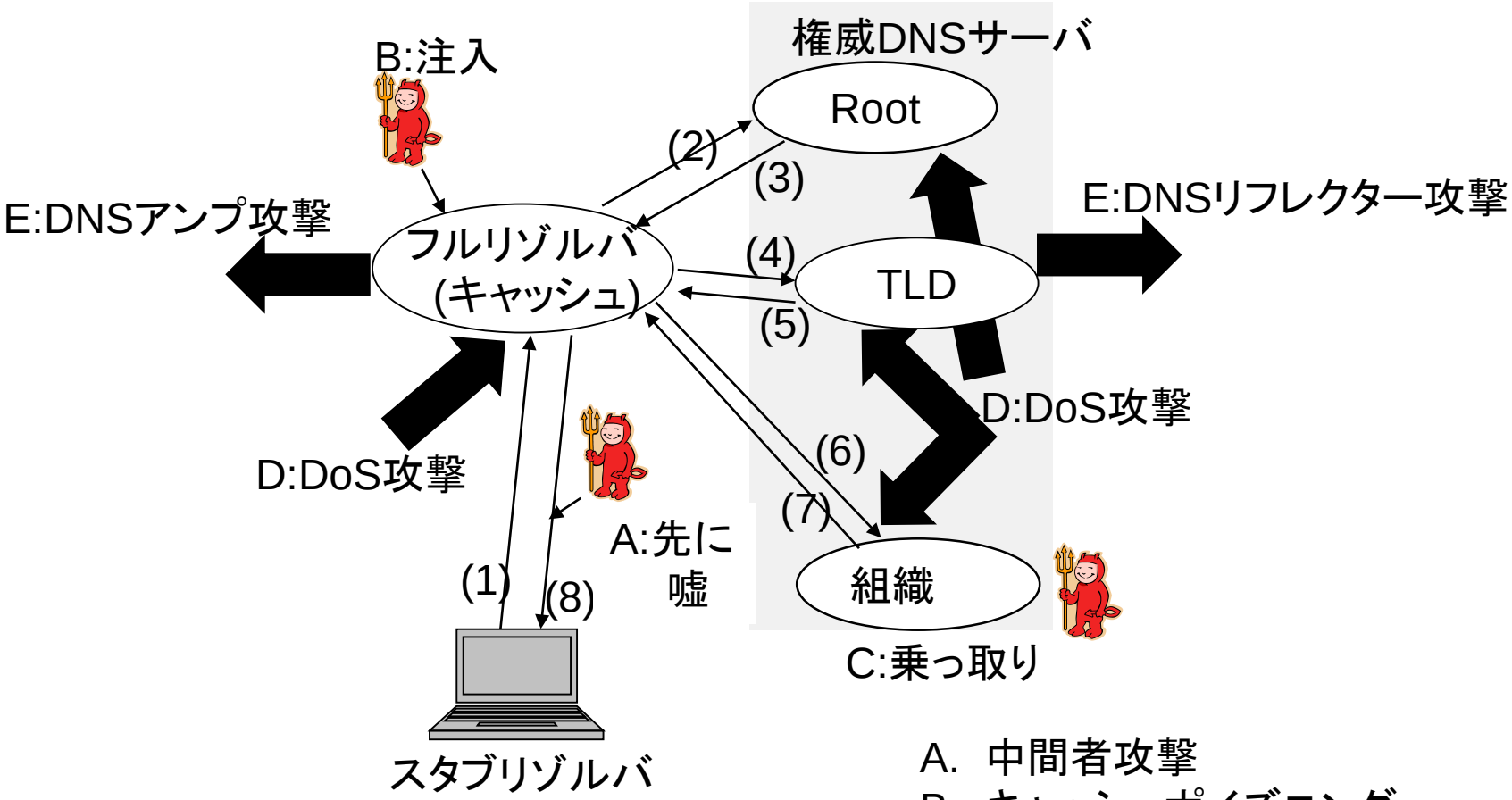
Copyright © 2014 Japan Registry Services Co., Ltd.

注意

- いわゆるキャッシュサーバという言葉は誤用なので、フルリゾルバに統一します
  - PowerDNSやBundy権威DNSサーバは権威DNSサーバだがキャッシュをサービスに使用
  - RFC 1035では明確ではないが以下の記載あり
    - 名前解決する機能をresolverとし、full resolver、resolver+cache、Recursive server+Central cacheの三通りの書き方で説明
  - RFC 1123ではstub resolverとa full service resolverを区別して紹介
  - Full service resolverかFull resolverが正しい

Copyright © 2014 Japan Registry Services Co., Ltd.

DNSへの攻撃の分類



RFC 3833 Threat Analysis of
the Domain Name System (DNS)

A. 中間者攻撃
B. キャッシュポイズニング
C. 権威DNSサーバの乗っ取り
D. サービス不能(DoS)攻撃
E. DNSリフレクター攻撃

Copyright © 2014 Japan Registry Services Co., Ltd.

A:中間者攻撃

- エンドユーザを偽サイトに誘導して利益を得るもの

- RFC 3833 Section 2.1. Packet Interception
- Wi-Fi区間やワイヤタッピングなどでクエリを監視し、
- 応答を書き換えたいクエリが来たら、
- 本物の応答より前に注入したい応答を返す
  - クエリの全情報を使えるので、確実に入る
  - TCPでも防げない
  - エンドノードでのDNSSEC検証やTSIGで検知可能
- 実装例: いくつかの国で実装されている事例あり
- 対策
  - 信用できないネットワークを使わない
  - 通信路の暗号化 (信頼できるところまでVPN)
  - TSIG/DNSSEC検証など

Copyright © 2014 Japan Registry Services Co., Ltd.

B:キャッシュポイズニング

- エンドユーザを偽サイトに誘導して利益を得るもの
- ID Guessing and Query Prediction (RFC 3833 2.2)
  - 権威DNSサーバからの応答を偽装し、フルリゾルバのキャッシュに偽の情報を注入するもの
  - GuessとかPredictionだが、乱数も推定の一つ
  - 短時間に注入するもの: Kaminsky attackなど
  - 長期にわたってゆっくり試行するもの: Birthday attack
- Name Chaining (RFC 3833 2.3)
  - NSやCNAME, MXなどのRDATAに含まれるドメイン名の情報に変な情報を書いて注入
  - 内部名ではなく権威がない名前のグルーなど
    - example.com IN NS ns.example.jp
    - ns.example.jp IN A 192.0.2.1 ← 他ドメイン名のグルー
  - いまどきのフルリゾルバは受け付けない

Copyright © 2014 Japan Registry Services Co., Ltd.

C:権威DNSサーバ乗っ取りなど

- エンドユーザを偽サイトに誘導して利益を得るもの

- 権威DNSサーバそのものの乗っ取り
  - 普通のホストの脆弱性経由
- ドメイン名登録情報の改竄
  - 登録者パスワードの脆弱性やパスワードリカバリ
  - レジストリやレジストラ、リセラのシステムの脆弱性

- 任意の誘導ができるが、
  DNSプロトコルの脆弱性ではない

Copyright © 2014 Japan Registry Services Co., Ltd.

D:サービス不能攻撃(DoS)

- DNSサーバに大量のクエリを送って負荷を上げ、サービスを妨害するもの
- DNSサーバのネットワークに大量のパケットを送って通信を妨害するもの

- 対策
  - 上位ネットワークにフィルタ依頼
    - IP Tracebackして発信元ASに対策を依頼
  - 忍耐 → 強化のための回線増強
  - IP anycastでDNSサーバ数を増やす
  - Rate Limit
    - DNS Response Rate LimitingしてDNSサーバの応答を減らす
  - 送信元を減らす活動 (BCP38やbotnet対策など)

Copyright © 2014 Japan Registry Services Co., Ltd.

E:DNSリフレクター攻撃の加害者

- 概要
  - DNSはUDPを主に使うため、IPアドレスの詐称に弱い
  - 送信元IPアドレスを攻撃先としたクエリをフルリゾルバや権威DNSサーバに送ると、攻撃先に応答を返す
  - DNSではクエリよりも応答のサイズが大きい
    - EDNS0で4000バイト程度まで返せる
    - DNSSECでクエリの20倍のサイズの応答を返す
- 対策
  - オープンリゾルバの停止
  - 権威DNSサーバではDNS Response Rate Limitting実装
  - Network Ingress filtering (BCP38)の実装

Copyright © 2014 Japan Registry Services Co., Ltd.

Node re-delegation attack

委任注入攻撃

委任注入攻撃

- Bernhard Müller, “Improved DNS spoofing using node re-delegation”, August 2008
- Wikipedia DNS spoofing
  - http://en.wikipedia.org/wiki/DNS_spoofing

Redirect the target domain's nameserver

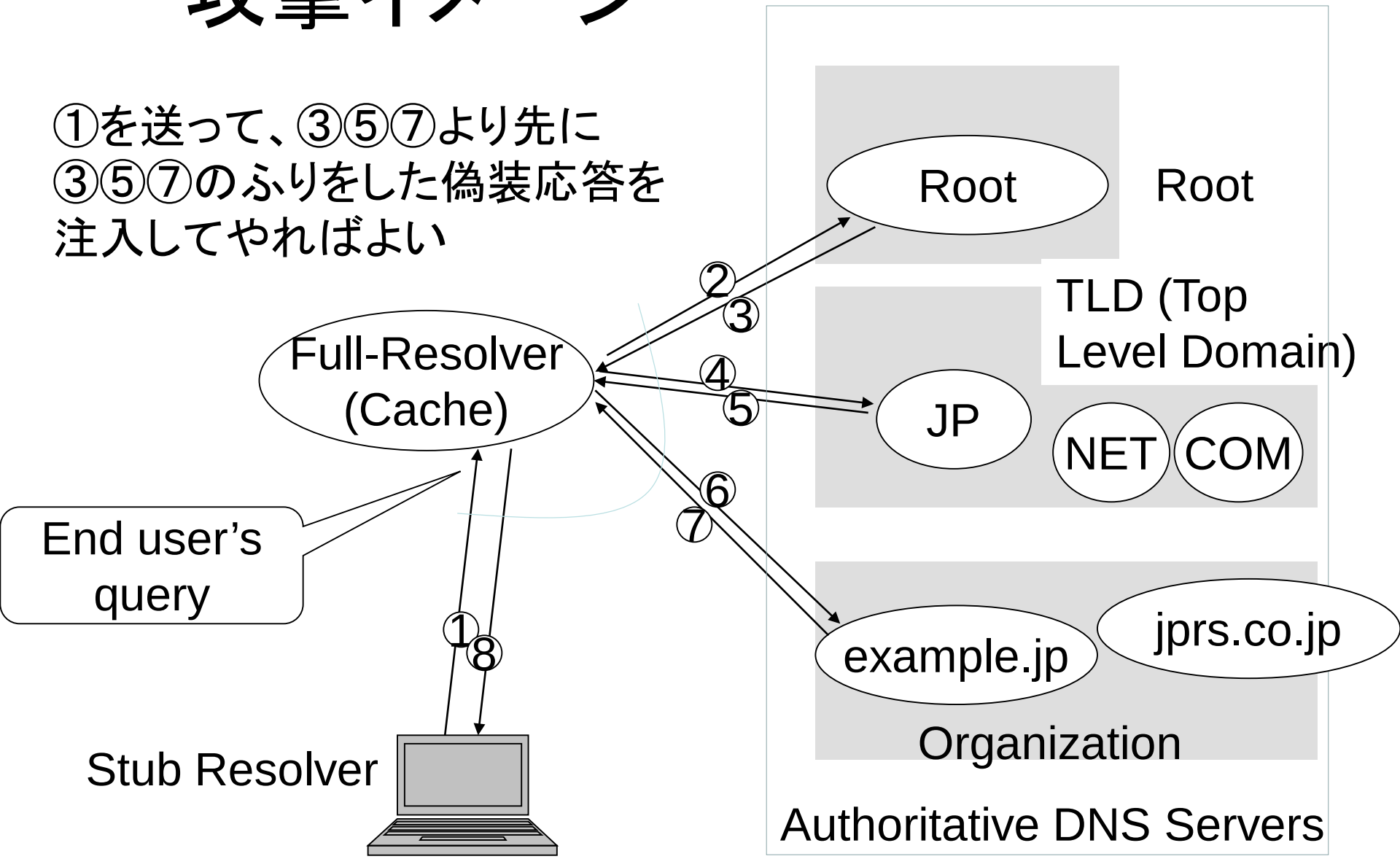
The first variant of DNS cache poisoning involves redirecting the nameserver of the attacker’s domain to the nameserver of the target domain, then assigning that nameserver an IP address specified by the attacker.

- Wikipediaにも書かれるぐらい、よく知られた攻撃手法
- 基本的には、Kaminsky攻撃の変形で、注入するものが委任情報

Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃イメージ

①を送って、③⑤⑦より先に
③⑤⑦のふりをした偽装応答を
注入してやればよい



Copyright © 2014 Japan Registry Services Co., Ltd.

Kaminsky型攻撃の特徴

- 攻撃対象の名前の前にランダムなラベルをつけたクエリ名を使用
  - 毎回ラベルが異なるため、フルリゾルバは権威DNSサーバへ毎回クエリを送る
  - 攻撃対象を管理する権威DNSサーバーから返る正当な応答は名前エラー(NXDOMAIN)
  - 攻撃失敗(正当な応答が先に到達)の場合でも、クエリ名のエラー情報だけがキャッシュされる
    - 攻撃対象の注入には影響しない
  - このため、一度攻撃に失敗してもランダムなラベルを変更することで、連続して攻撃できる

Copyright © 2014 Japan Registry Services Co., Ltd.

委任注入攻撃
node re-delegation attack

例：www.example.jp というホスト名を注入する場合

1. (random).www.example.jp Aクエリを攻撃対象に送る
   - example.jpを管理する権威DNSサーバは名前エラーを返す
   - 偽造応答の注入が成功しなかった場合、
     - (random).www.example.jpのエラーのみキャッシュされる
     - www.example.jpの情報はキャッシュされない
2. 偽造応答”www.example.jp IN NS *my-server*”をexample.jpのサーバのアドレスから注入する
   - my-serverにwww.example.jpの偽ゾーンを事前準備しておく

実際の攻撃では、ID(問合せと応答を紐付ける番号)のみを変更した、多数の偽造応答を同時に送る

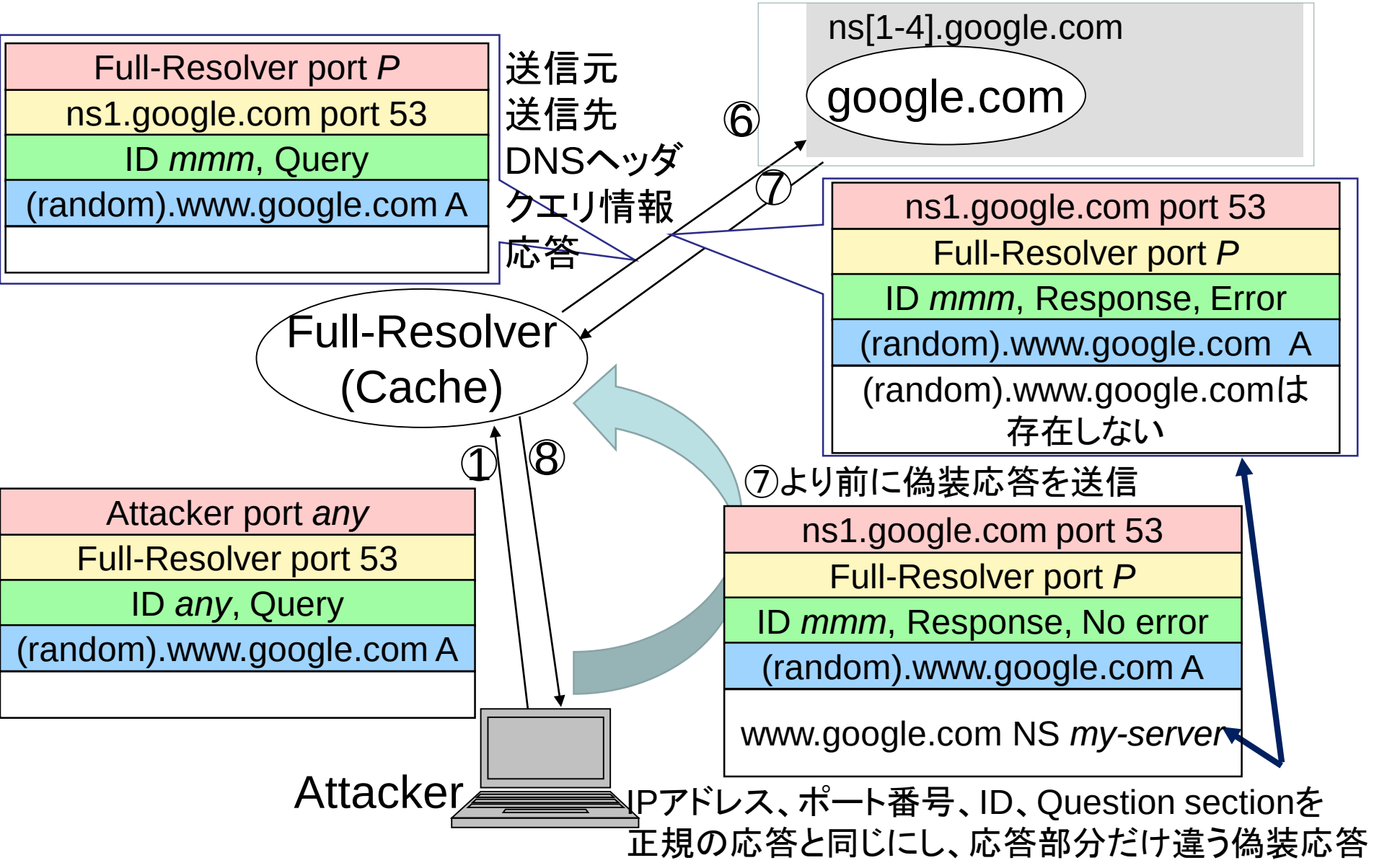
Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃の詳細

- トリガードメイン名の選定 ($Trigger)
- トリガークエリ “(*random*).$Trigger A” をフルリゾルバに送信
  - フルリゾルバは最終的に権威をもつ権威DNSサーバにクエリを送り、権威DNSサーバは(random).$Triggerの名前エラーを返す
  - $Trigger自体はキャッシュされない
- 偽装応答をフルリゾルバに送信
  - IP src = $Triggerの権威DNSサーバアドレス
  - IP dst = フルリゾルバのアドレス
  - src port = 53
  - dst port = フルリゾルバのポート番号 (固定か乱数)
  - DNS header: Rcode 0, response, ID = 乱数
  - Question section = “(*random*).$Trigger A”
  - Authority = 注入したいもの: “Target NS my-server”
- 事前にmy-serverにTargetゾーンを用意しておくこと
  - Target IN NS my-server
  - Target IN A 192.2.0.2
  - *.Target IN A 192.2.0.2

Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃例: www.google.com



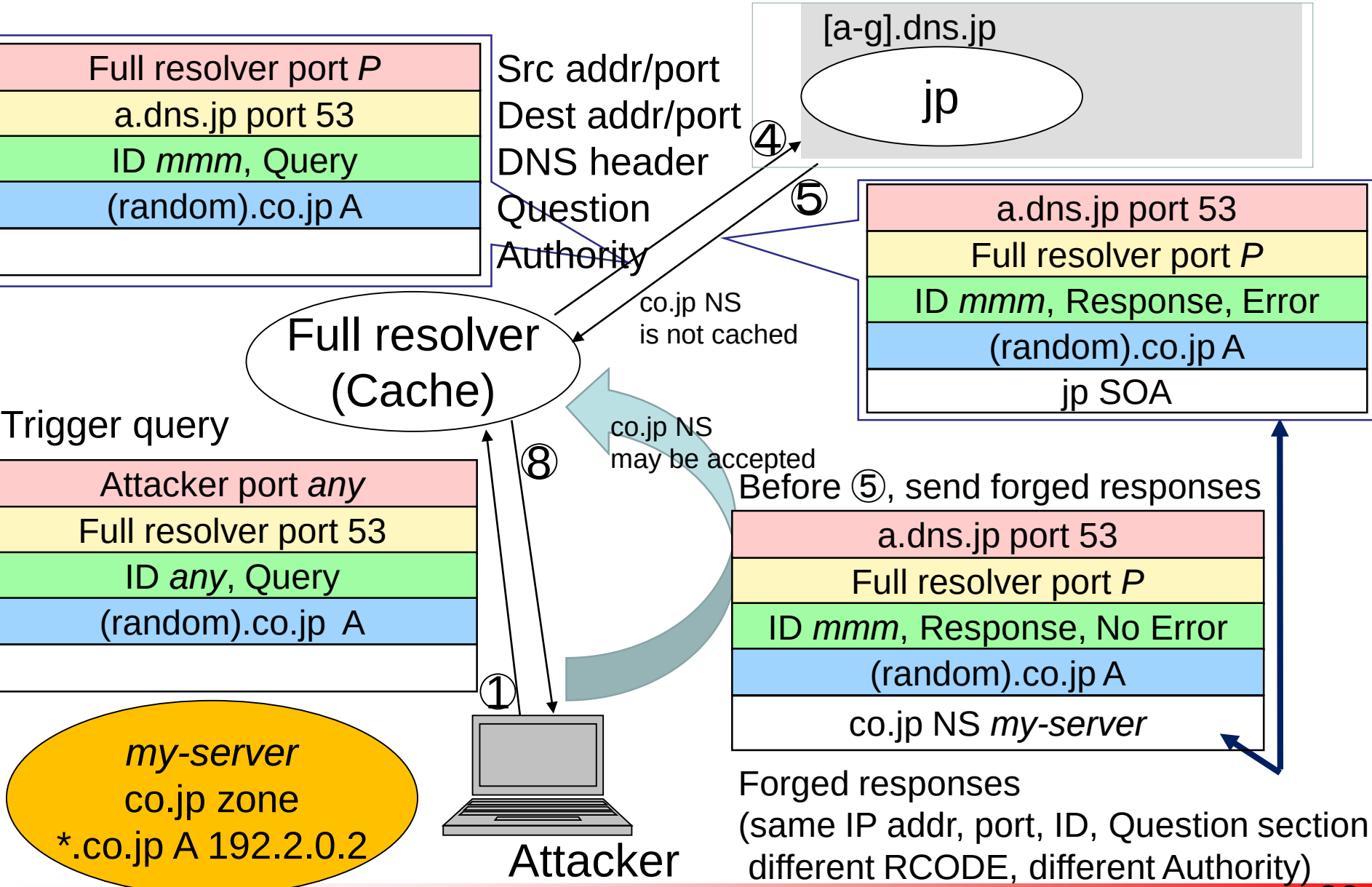
Copyright © 2014 Japan Registry Services Co., Ltd.

注入が成功しうるドメイン名

1. NSを持たないすべてのドメイン名
   - ふつうのホスト名(www.google.com, www.ieice.org, ...)
   - いくつかの中間ドメイン名 (co.jp, tokyo.jp, …)
2. いくつかの委任 (NSがあるドメイン名)
   - あるドメイン名と、その子孫が同一DNSサーバに共存
   - 最上位のドメイン名は対象外
   - 最下位の委任は対象外
   - 例1: “net” and “root-servers.net”
     - ルートDNSサーバは “.” と “root-servers.net” ゾーンを保持
   - 例2: “co.uk”
     - uk DNSサーバは “uk” ゾーンと “co.uk” ゾーンを保持
   - 重なりが親子の場合、親から子への委任を示すNSリソースレコードはキャッシュされないので注入が容易
     - uk DNSサーバは、co.ukドメイン名のクエリに対して、co.ukの委任ではなくラベル.co.ukを返す
     - 実環境でのnetの注入は困難

Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃例:co.jp



Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃例:net

Only when "net NS" is not cached by victim full resolvers

Full resolver port *P*	Src addr/port
a.root-servers.net port 53	Dest addr/port
ID *mmm*, Query	DNS header
(random).root-servers.net A	Question
	Authority

[a-m].root-servers.net

② "." and root-servers.net

③

a.root-servers.net port 53
Full resolver port *P*
ID *mmm*, Response, Error
(random).root-servers.net A
root-servers.net SOA

net NS
is not cached

Full resolver
(Cache)

Trigger query

Attacker port *any*
Full resolver port 53
ID *any*, Query
(random).root-servers.net A

⑧

net NS
may be accepted

Before ③, send forged responses

a.root-servers.net port 53
Full resolver port *P*
ID *mmm*, Response, No Error
(random).root-servers.net A
net NS *my-server*

①

my-server
net zone
*.net A 192.2.0.2

Attacker

Forged responses
(same IP addr, port, ID, Question section
different RCODE, different Authority)

Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃例:root-servers.net

~~Only when "net NS" is not cached by victim full resolvers~~

[a-m].root-servers.net

"." and root-servers.net

Full resolver port *P*
a.root-servers.net port 53
ID *mmm*, Query
(random).root-servers.net A

Src addr/port
Dest addr/port
DNS header
Question
Authority

②

③

a.root-servers.net port 53
Full resolver port *P*
ID *mmm*, Response, Error
(random).root-servers.net A
root-servers.net SOA

root-servers.net NS
is not cached

Full resolver
(Cache)

Trigger query

Attacker port *any*
Full resolver port 53
ID *any*, Query
(random).root-servers.net A

⑧

root-servers.net NS
may be accepted

Before ③, send forged responses

a.root-servers.net port 53
Full resolver port *P*
ID *mmm*, Response, No Error
(random).root-servers.net A
root-servers.net NS *my-server*

①

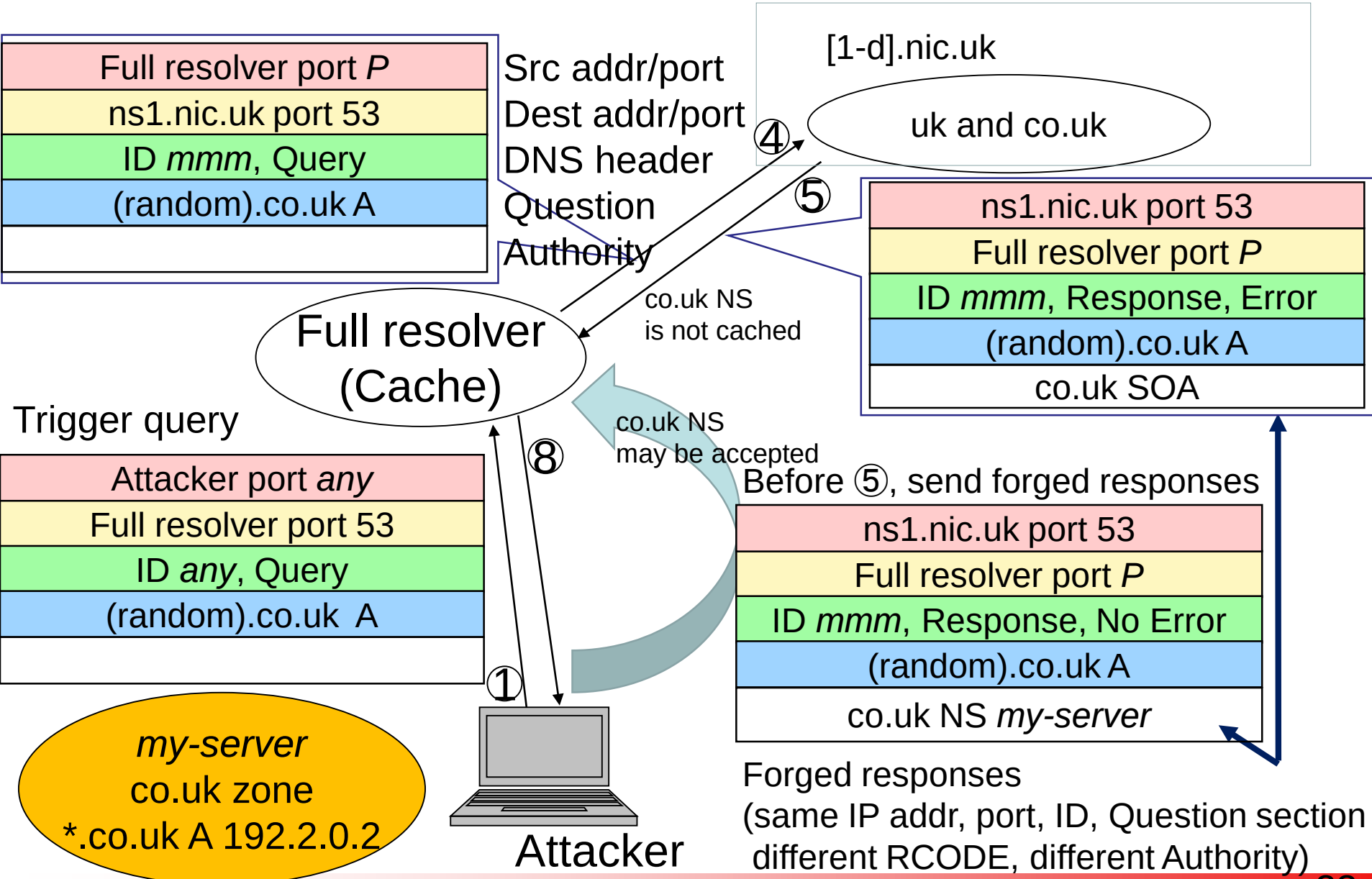
my-server
root-servers.net zone
***.root-servers.net A**

Attacker

Forged responses
(same IP addr, port, ID, Question section
different RCODE, different Authority)

Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃例:co.uk



Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃成功確率と期待値

	変数	単位	数値の範囲	例
フルリゾルバのポート数	$Nport$		$1 \sim 2^{16}$	1 or 64,000
QID数	$Nqid$		2^{16}	65,536
権威サーバアドレス数	Nns		$1 \sim 13$	4
権威サーバへのRTT	$Tauth$	秒	$0.001 \sim 0.2$	0.039
繰り返し時間	$Tloop$	秒		0.020
偽応答数の送出レート	$Rans$	パケット/秒		100,000

- 一度目の試行で入る確率

$$P = \frac{Tauth * Rans}{Nqid * Nport * Nns}$$

Example:
0.0148
Or
0.00000023

- 連続攻撃時に入る確率 1
- 攻撃にかかる時間の期待値

$$T = \frac{Nqid * Nport * Nns}{Rans}$$

Example:
2.62 seconds
Or
167,772 (2 days)

Copyright © 2014 Japan Registry Services Co., Ltd.

一度目の試行で入る確率

- ID, port, アドレスが一致すると確実に入るとする
- 一度の試行($Tauth$時間)で入る確率 P

$$P = \frac{Tauth * Rans}{Nqid * Nport * Nns}$$

- ただし$Tauth$は制御できないので
  $Tloop < Tauth$という条件で$Tloop$を用いる

$$P = \frac{Tloop * Rans}{Nqid * Nport * Nns}$$

Copyright © 2014 Japan Registry Services Co., Ltd.

連続攻撃時に入る確率

- $n-1$回目までに入らなくてn回目に入る確率

$$Pn = (1-P)^{n-1} * P \qquad (1)$$

- n回目までに入る確率QnはPnの和

$$Qn = \sum_{i=1}^{n}(1-P)^{i-1}P \qquad (2)$$

- 変形すると

$$Qn = 1-(1-P)^{n} \qquad (3)$$

- nを無限大に近づけると

$$Qn \to 1$$

Copyright © 2014 Japan Registry Services Co., Ltd.

連続攻撃時に入る回数の期待値

- $n-1$回目までに入らなくてn回目に入る確率

$$Pn = (1-P)^{n-1} * P \qquad (1)$$

- n回目までに入る期待値Enは回数*確率の和

$$En = \sum_{i=1}^{n} i(1-P)^{i-1}P \qquad (2)$$

- 変形すると

$$En = \frac{1}{P} - \frac{(1+nP)(1-P)^n}{P} \qquad (3)$$

- nを無限大に近づけると

$$En \to \frac{1}{P}$$

成功する場合の期待値なので$0 < P \leq 1$
Enの(3)右辺の第二項は常に正
かつ(2)よりEnは単純増加なので、
Enは$1/P$を上界とする単調増加数列

Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃にかかる時間の期待値

$$T = Tloop * E = \frac{Tloop}{P}$$
$$= \frac{Nqid * Nport * Nns}{Rans}$$

- サーバ数4とし、100000ppsで偽装応答を送ると
- Port randomizationしていない場合
  65536*1*4/100000 = 2.62秒
- Port randomizationしていると
  65536*64000*4/100000= 167772.16秒(約2日)
- 2日間にわたって10万ppsの怪しいパケットが来れば気がつくでしょう

Copyright © 2014 Japan Registry Services Co., Ltd.

ソースポートランダマイゼーション

- 攻撃にかかる時間の期待値

$$T = \frac{Nqid * Nport * Nns}{Rans}$$

- Tを大きくするには、$Nport$を2^{16}にすること
- ポート番号を16ビット使い、クエリごとにランダムにすること
- これをソースポートランダマイゼーションという
- 以下で実装
  - DJBDNS dnscache (~2001)
  - PowerDNS recursor (2006/4リリース)
  - BIND 9 (9.5.0-P1, 2008年5月~)
  - 2008年以降の実装

Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃ツール試作

- Cで500行、標準ライブラリのみ使用
  - selectでタイミングと送受信の管理
  - raw socketで偽造レスポンス送出
- 与えるパラメータ
  - 攻撃対象のフルリゾルバのIPアドレス、ポート番号
  - トリガードメイン名 (連続攻撃向けのドメイン名)
  - 注入したいNS RRのドメイン名、サーバ名
  - トリガードメイン名の権威サーバのIPアドレスリスト
- 攻撃例
  - ./a.out 192.2.0.2 20001 www.google.com www.google.com ns.dnslab.jp 216.239.32.10/216.239.34.10/216.239.36.10/216.239.38.10

Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃ツールの構造

初期化
- socket 2個作成
  - 普通のUDPとRaw
- タイマー初期化
- 初回のクエリ送信

Loop
- select
- 応答を受けたら、注入が成功したか判定
  - 成功したら終了
  - 失敗したら、トリガークエリ送信へ
- Tloop時間たったら、トリガークエリ送信へ
- Raw socketにかけたら、偽造応答送信へ

終了
- 成果の表示

トリガークエリ送信
- (random)を生成
- 攻撃対象のport 53へ
- クエリを送信

自分のアドレス port *any*
Full-Resolver port 53
ID *any*, Query
(random).トリガードメイン名 A

偽造応答送信
- IDをランダムに生成
- 権威サーバアドレスをランダムに選択
  - 引数であたえたアドレスリストより
- 偽造応答を生成
  - Question Sectionはクエリ送信と同じ
  - Authority Sectionに注入するNS RR
- 権威サーバのport 53から
- 攻撃対象の指定ポートへ
- Raw socketで送信

権威サーバアドレス port 53
Full-Resolver port *P*
ID *random*, Response
(random).トリガードメイン名 A
注入するNSリソースレコード

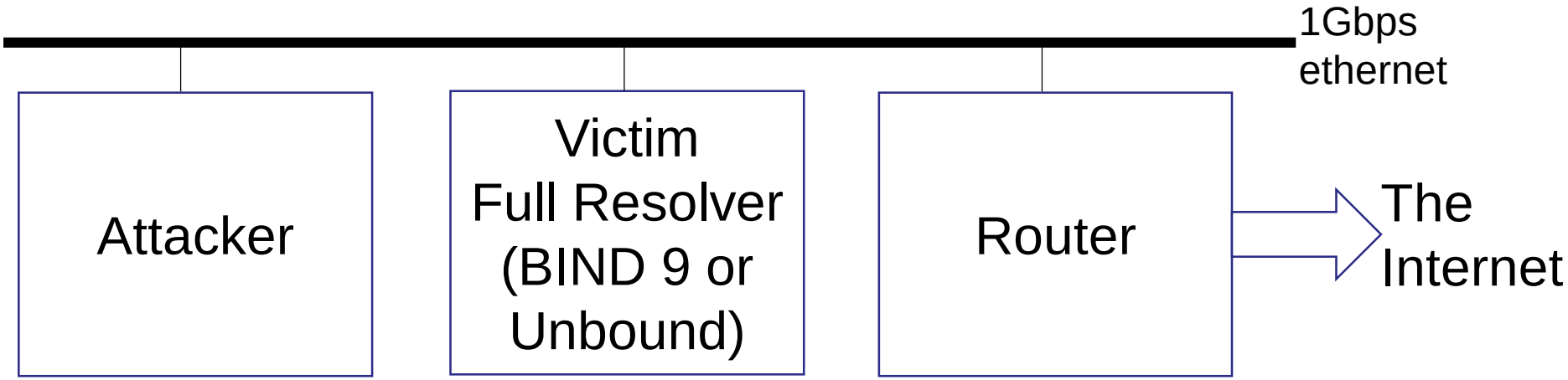
Copyright © 2014 Japan Registry Services Co., Ltd.

Tips

- Raw socketでのbyte order
  - FreeBSD: The ip_len and ip_off fields must be provided in host byte order. All other fields must be provided in network byte order.
  - BSD以外はすべてnetwork byte orderのはず
- 終了判定方法
  - 誘導先権威DNSサーバに *.domainname A を書いておくことで、トリガーとなるランダムクエリに存在する応答が戻る
- デバッグツール
  - tcpdump
  - rndc dumpdb

Copyright © 2014 Japan Registry Services Co., Ltd.

実験環境



- 構成
  - ~~箱庭を準備するのは面倒なので~~
  - より現実に近い評価にするために
  - JPRSの研究ネットワークでグローバルアドレスを使って実施
  - 攻撃対象の名前解決クエリが外に漏れるが気にしない
- Two VMs (Attacker and Victim)
  - 2.5GHz Xeon, 2 cpu, 3GB memory
- Victim full resolvers
  - BIND 9.9.5 (without source port randomization)
  - Unbound 1.4.21 (without source port randomization)

Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃の前提と実験の条件

- 前提: 攻撃ツール以外からのクエリなし
  - 本物が入ったらTTL時間は攻撃成功しにくいため
- ソースポートランダマイゼーションをoffにしたBIND 9, Unboundを用意
  - query-source 192.0.2.2 port 20001; (BIND 9)
- 誘導先のDNSサーバとしてns.dnslab.jp
  - 実験で攻撃してみたゾーンが多数書いてある
  - *.domainname Aを書いておく(終了判定のため)
- 簡略化のためIPv4でのみ評価

Copyright © 2014 Japan Registry Services Co., Ltd.

実験結果

- 注入可能なドメイン名を注入成功
  - NSリソースレコードが存在しないもの
    - www.google.com
    - co.jp go.jp lg.jp chiyoda.lg.jp
    - tokyo.jp saitama.jp kanagawa.jp chiyoda.tokyo.jp
  - 先祖と子孫の同居
    - co.uk
    - root-serers.net
    - net (事前にrndc flush)

- 入らないときはrndc dumpdbして確認
  - 正規のAと、偽造NSが混ざることもある
  - AがexpireするとNSが有効に使用される

Copyright © 2014 Japan Registry Services Co., Ltd.

期待値と攻撃結果の比較

	変数	単位	変数の範囲	攻撃対象 www.google.com		攻撃対象 net	
Port randomization				no	yes	no	yes
フルリゾルバポート数	Nport		1~2^16	1	64000	1	64000
ID数	Nqid		2^16	65536	←	65536	←
権威サーバアドレス数	Nns		1~13	4	←	13	←
権威サーバへのRTT	Nauth	秒	0.001~0.2	0.036~0.094	←	0.006~0.272	←
繰り返し時間	Nloop	秒		0.036	←	0.033	←
偽装応答の送出レート	Rans	pps		66726	←	69158	←
一度目攻撃成功確率				0.00916	0.00000014	0.00267	.00000004
攻撃成功の期待値		秒		3.9	251434	12.3	788425
攻撃時間の実測値		秒		46	×	8	×

Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃時間を短縮する最適化

$$T = \frac{Nqid * Nport * Nns}{Rans}$$

- 分母の偽造応答の送出レートを大きくする
  - $Tloop < Tauth$に注意
- 例: 偽装応答を1Gbpsで送ると応答パケットサイズ100の場合、1,000,000pps
- ポートランダマイゼーションしていない場合

65536 * 1 * 4 / 1000000 = 0.262秒　瞬殺

- ポートランダマイゼーションしている場合

65536 * 64000 * 4 / 1000000 = 16777　4時間半

Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃手法のまとめ

- ソースポートランダマイゼーションを有効にしていないと数秒から数分で注入できる
- ターゲットはほとんどのドメイン名
  - A, AAAA, PTRなどがついているほとんどのホスト名
  - いくつかの中間のドメイン名
- ソースポートランダマイゼーションしていると数時間では入らないが、性能向上すれば可能になる
  - random.www.google.comやrandom.root-servers.netという怪しいクエリを出し続けるのはいやなので止めた
- 成功しないときは攻撃対象のcacheをflushする
  - net, root-servers.netへの攻撃は、キャッシュにnet, root-servers.net NSがあると成功しない
  - www.google.com Aがあっても、www.google.com NSは入ることに注意

Copyright © 2014 Japan Registry Services Co., Ltd.

試作した手法の欠点

- 攻撃が検知されやすい
  - 送ってない応答が大量に到達するので、入出力のパケット数を見るだけでわかる
  - トリガークエリが多めなことと、送っていない応答を捨てる処理のためにフルリゾルバの負荷が上がる
- 攻撃者を特定しやすい
  - 誘導先のDNSサーバを用意する必要がある
  - トリガーとなるクエリを送り、応答を確認するために、そのIPアドレスを詐称できない
- 攻撃側にもリソースが必要
  - 帯域と、高性能機器

Copyright © 2014 Japan Registry Services Co., Ltd.

委任注入攻撃への対策

- サービス提供者はDNSだけに依存せずに、他の検知技術と組み合わせること
  - 証明書によるSSL/TLS (HTTPS) など

- DNSSEC検証による攻撃検知
- フルリゾリバ運用者の対策
  - 長期的な対策
  - 注入を検知した場合の対策
- フルリゾルバ開発者の対策
- 権威DNSサーバ運用者の対策

Copyright © 2014 Japan Registry Services Co., Ltd.

DNSSEC検証による攻撃検知

- 以下、BIND 9, Unboundで確認
- DNSSEC検証していてもキャッシュには入る
- DNSSEC検証により攻撃検知可能
  - スタブリゾルバにはServer failureを返す
  - netにNSを注入できても、net DNSKEYを検証できないため、検証エラー
- TLDは概ねDNSSEC対応
  - ただし複雑な構造を持つTLDでNSEC3 OptOutしている場合に、検証できない中間ドメイン名ができる場合がある
  - 2014年3月にJPRSはJPゾーン内のすべての中間ドメイン名に意味のないTXTリソースレコードを追加して対応するNSEC3リソースレコードを生成させるようにした

Copyright © 2014 Japan Registry Services Co., Ltd.

NSEC3により検知可能な理由

- kanagawa.jp には対応するNSEC3 RRあり
  - ONVMIMIF2S8PQSOHC68NDN6805PU97II.jp. 900 IN NSEC3 1 1 5 66637984FE OO12LTU5GEAH5UB3SO8ASTEL11B0K86T TXT RRSIG
  - タイプビットマップにはTXTとRRSIG
- ここでkanagawa.jp NSを注入できたとする
- example.kanagawa.jp を検証する時には、kanagawa.jp NSがあるので、kanagawa.jp DSがあるかどうかを調べる
  - RFC 4035 Section 5.2
  - kanagawa.jp DSクエリを送るとNSEC3がもどる
  - NSEC3より、kanagawa.jpにNS, DSなし
  - 矛盾となるため、検証エラーとなる

Copyright © 2014 Japan Registry Services Co., Ltd.

フルリゾルバ運用者の対策 (1)

- ソースポートランダマイゼーション
  - 時間の猶予が得られる
- Unbound: harden-referral-path optionの有効化
- 監視
  - フルリゾルバの負荷
    - 多量の不存在クエリ、知らない応答
  - 入出力のパケット数
    - DNSはクエリ、レスポンスがほぼ一対一なので、非対称なパケット数は攻撃の可能性あり
  - 追記: 空けてないポートへのUDP (応答のICMP)
  - 監視サービス
    - フルリゾルバ・権威サーバ間の通信の監視サービスあり
    - Farsight Security, Inc., Nominum, Inc.
    - 応答を過去のログと比較

Copyright © 2014 Japan Registry Services Co., Ltd.

フルリゾルバ運用者の対策 (2)

- キャッシュポイズニングに気がついたら私に教えてください、事例が欲しい

- 注入されたRRSetの消し方
  - 注入されたRRSetをキャッシュから消す
    - rndc flushname 攻撃されたドメイン名
  - 注入されたドメイン名を問い合わせ
    - dig @フルリゾルバ 攻撃されたドメイン名 A など
  - 他のフルリゾルバとの応答を比較
    - 自分が管理しているサーバかPublic serviceと比較
    - キャッシュから消すという行為で注入の可能性が上がるためで、複数のフルリゾルバに同時に注入するのは困難なため他との比較が有効である
    - 特に有名なPublic serviceは対策しているはず

Copyright © 2014 Japan Registry Services Co., Ltd.

フルリゾルバ開発者の対策 (1)

- ソースポートランダマイゼーションの実装: 完了
- 監視機能の追加
  - フルリゾルバは送っていない応答を捨てるが、それを用いて警告する
  - 送っていない応答内に、攻撃者が注入しようとしているものがある (委任注入攻撃のときはNS RRSet)
  - ランダムクエリから、共通部の抽出
  - 量が多いため、集約して量を減らすこと
- 対策: TCP transportの使用
  - 権威サーバとの間の通信をTCPにするとこのツールでは攻撃できない
  - シーケンス番号が32ビットなので、他のパラメータとあわせて64ビットの推定が必要となり、耐性が高い
  - ただし、TCP通信は重い (ステートと、RTTが3倍)

Copyright © 2014 Japan Registry Services Co., Ltd.

フルリゾルバ開発者の対策 (2)

- 検知とTCPによる対策を組み合わせると効果的
  - キャッシュポイズニング攻撃を受けているドメイン名だけTCPで通信
  - 既に一部のフルリゾルバに実装されている
  - draft-fujiwara-dnsop-poisoning-measures-00 を書き、問題提起した
    - 実装されていない実装に実装してもらうため
- DNS CookieでID数を増やす(dnsopで停滞)
  - ISCはBIND 9.10で実験的に実装 (Source Identity Token, SIT)
- Unboundではharden-referral-pathを実装
  - ルート,TLDへreferralを取り直すため、クエリ数が増える
- Google Public DNS: nonce prefix実装, 多数のソースIPアドレスの使用

Copyright © 2014 Japan Registry Services Co., Ltd.

権威DNSサーバでの対策

- NSがないドメイン名はすべて攻撃対象
- 子孫と同居しているとすべて攻撃対象

ということで、

　　一段ごとに完全にゾーン分割

　　一段ごとに権威DNSサーバを分離

するとある程度守れるのではないか

Copyright © 2014 Japan Registry Services Co., Ltd.

ホスト名の守り方

- ホスト名ごとにゾーンとDNSサーバアドレスを分ける
- ホスト名と同じアドレスをホスト名ゾーンのDNSサーバとする
  - example.jp ゾーン (192.168.0.2ではないアドレスで動かす)
    - www.example.jp IN NS www.example.jp
    - www.example.jp IN A 192.168.0.2 (グルー)
  - www.example.jp ゾーン (www.example.jp)
    - ゾーン頂点にA,AAAAを書く、複数可
    - www.example.jp IN A 192.168.0.2 (権威あり)
    - www.example.jp IN NS www.example.jp
- 問題点
  - サービスアドレスと、権威DNSサーバが同じアドレス
  - 変更時には上位のグルーと下位を同時に変更
  - 権威DNSサーバのアドレスを増やすと、サービスアドレスも増える
  - 内部名のDNSサーバ名には適用不可能
    - なぜなら、親子同居になるため
    - どこかには必ず内部名のDNSサーバが必須のため、どこかで負ける
  - RFC 2181 Ranking問題のバグを持つサーバへの注入は可能

Copyright © 2014 Japan Registry Services Co., Ltd.

中間ドメイン名の守りかた

- すべての階層をゾーンに分割
  - IPv6の逆引き　どうするんでしょうか？
    - f.e.d.c.b.a.9.8.7.6.5.4.3.2.1.0.f.e.d.c.b.a.9.8.8.b.d.0.1.0.0.2.ip6.arpa PTR
  - _sip._udp や selector._domainkeys も注意
    - SIP: _sip._udp.example.com SRV 0 1 5060 …..
    - DKIM: selector._domainkeys.example.com TXT

- 階層ごとに別サーバに分離
  - 階層ごとにIPアドレスとサーバが必要

Copyright © 2014 Japan Registry Services Co., Ltd.

JPでの中間ドメイン名対策の可能性

- 現在はすべてのJPドメイン名の委任を1ゾーンで管理
- 構造が楽しいぐらいに複雑、特に地域型とlg.jp
- 分離対象
  - 属性9、都道府県47*2: なんとかなる？
  - 地域型: 市町村区名ラベル 1300程度、新規登録なし
  - lg.jp: {union,vill,town,city,metro,pref}.ラベル.lg.jp
    - 都道府県市町村区名、事務組合名ごとに別ゾーン　1400
  - 合計約3000ゾーン
    - DNSSECの運用ができるか？
    - ゾーン転送が5分以内に完了するか？ (BIND 9にバグあり)
    - 運用コストは非常に大きくなりそう
- 分離作業の問題
  - DNSSEC検証の継続性 …. 不可能、移行時はJP DS削除
- 可能か？

Copyright © 2014 Japan Registry Services Co., Ltd.

中間ドメイン名 (dns.jp)

- 2014/6/24 JPRSはdns.jpをJP DNSから分離しました
  - IPアドレスも変更
  - 一つづつ分離ならば時間をかければ可能

- これで、dns.jpへの注入は困難になりました
  - ただし、a.dns.jpなど個々のホスト名への注入は可能

Copyright © 2014 Japan Registry Services Co., Ltd.

攻撃実績

- いまのところ、聞いたことはありません
- ご存知のかたがいらっしゃったら教えてください → (追記) DNS-OARC 2008 Workshop (Ottawa) にてPaul Vixieが示したようです

- DoSはよく聞くのに、キャッシュポイズニングの話は聞かない
- DoSのほうが現実の問題なのか？
- 費用対効果を考えた対策が重要

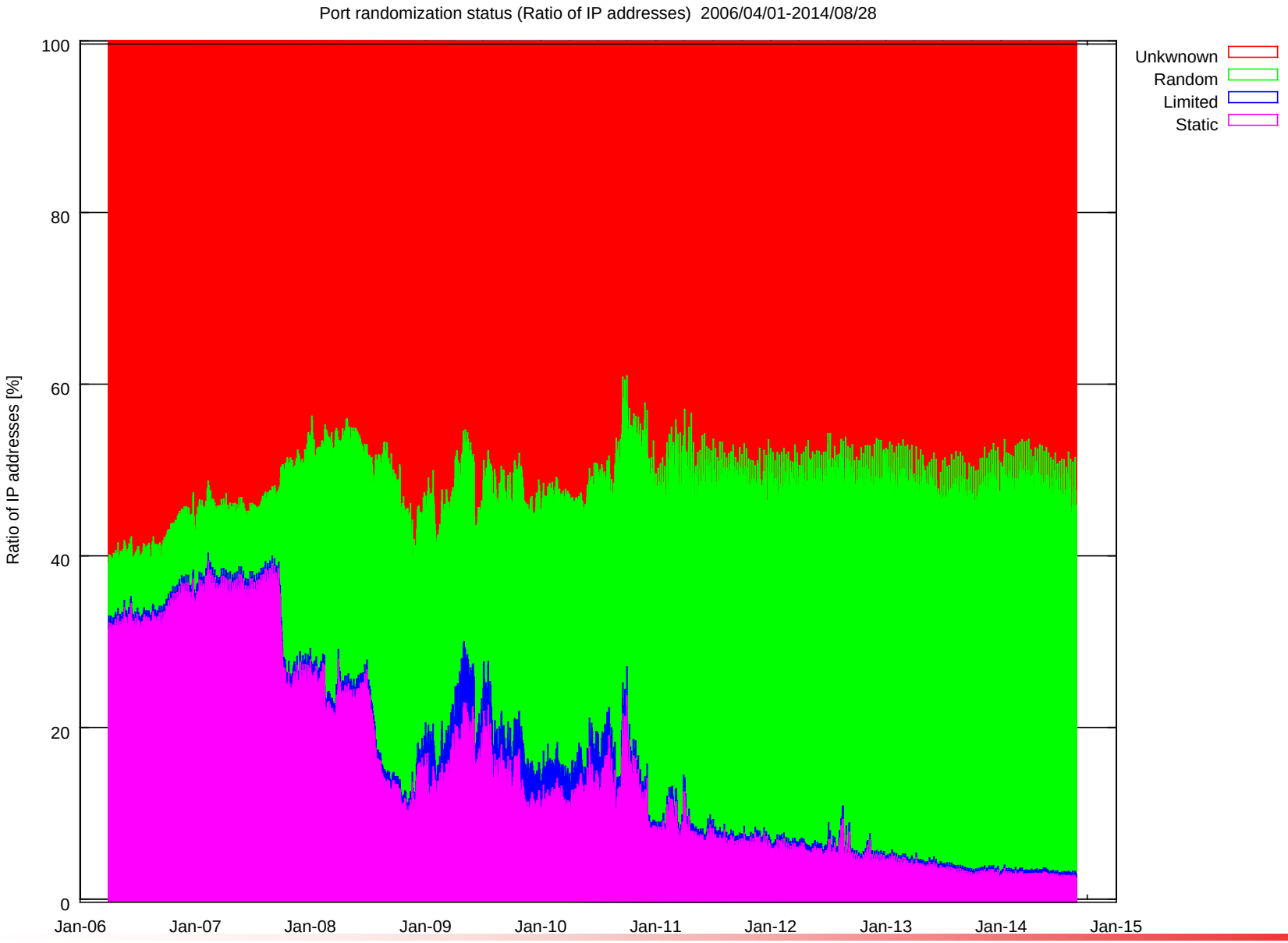
Copyright © 2014 Japan Registry Services Co., Ltd.

ソースポートランダマイゼーション (以下、SPR) 普及度調査

- JPRSでは2006年4月からのA.DNS.JP及びG.DNS.JPのクエリログを蓄積
- ソースポートランダマイゼーションの評価方法
  - 1日ごとのソースポートの使用数を数えたいが
  - ポート番号の上位下位8ビットの使用数を評価
- 判定基準
  - 1日10クエリ未満: 判定不能(Unknown)
  - 上位下位とも3以下: SPR無効 (Static, 固定)
  - 上位下位とも4以上: SPR有効 (Random)
  - 上位か下位のどちらかが3未満 (Limited, 限定)

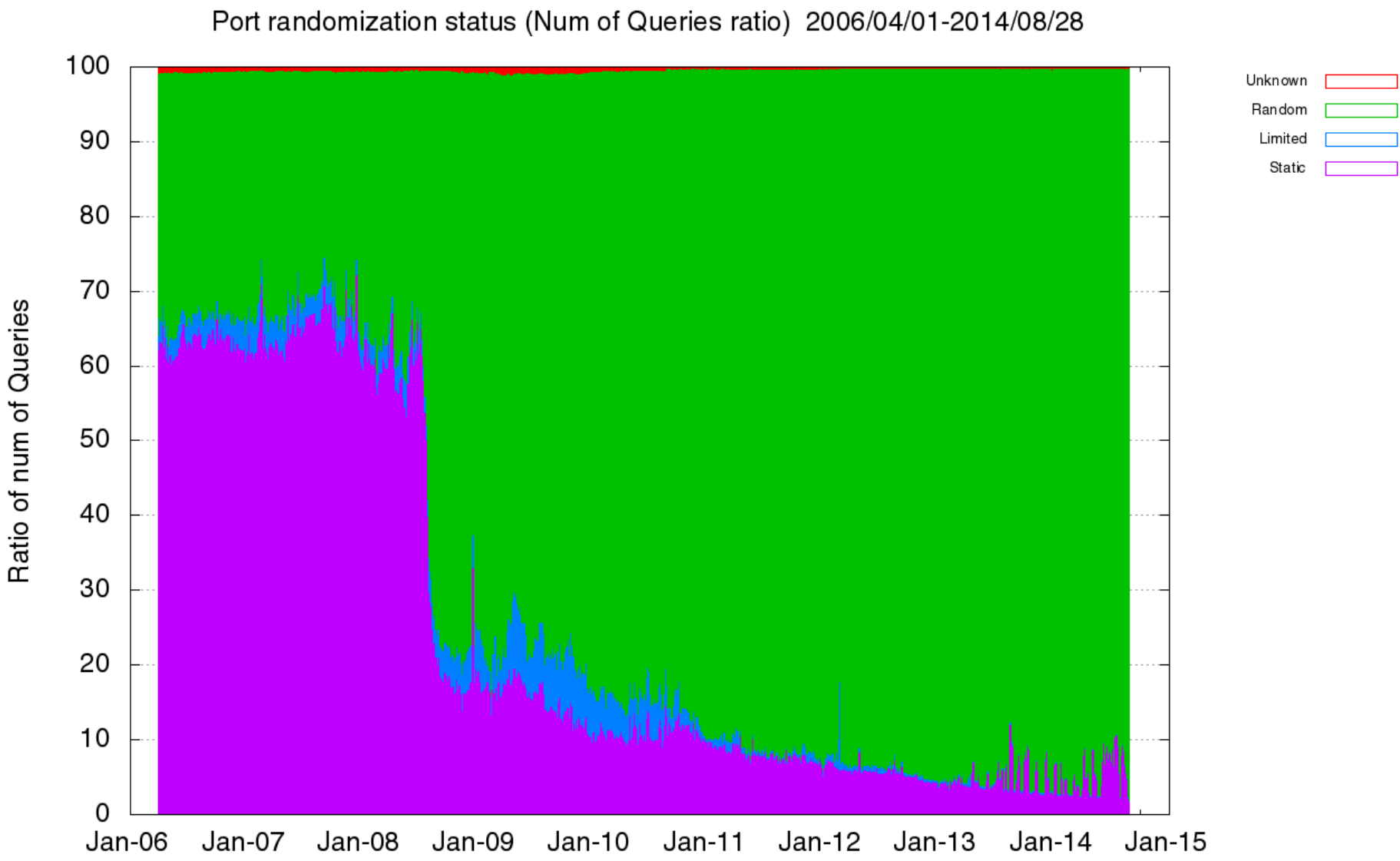
Copyright © 2014 Japan Registry Services Co., Ltd.

SPR普及度の変化



Copyright © 2014 Japan Registry Services Co., Ltd.

SPR対応度ごとのクエリの割合



Copyright © 2014 Japan Registry Services Co., Ltd.

SPR対応のまとめ

- 約半数のアドレスが判定不能
- Kaminsky攻撃が発表された2008年より前からSPRに対応していたアドレス　8%程度で2年間割合同じ
  - それらはDJBDNS dnscacheだと考えられる
  - 2007年後半よりSPRの実装が進んだこと
- いまでも4%程度のアドレスはSPR非対応
- Unknownを除くと判定可能なアドレスの約8%が非対応と考えられる
- 定常的にクエリを送るアドレスで固定なものからのクエリ量は減ってきている (3%以下)
- 大量にクエリを出す異常なアドレスは固定なものが多い？

Copyright © 2014 Japan Registry Services Co., Ltd.

IETF方面、開発者の反応

- IEPG meeting
  - 2014/7/20に報告を行ったが、内容が既知のため、興味を引かれず、フルリゾルバのIPアドレスを増やすとID数を増やして耐性を上げられるというコメントだけをもらった
- IETF dnsop
  - 2014/7/22にインターネットドラフトの発表を行ったが、対策の一つであるというコメントをもらっただけで、反応は薄い
  - DNS Cookieの標準化は停止中
- フルリゾルバ開発者は、対策を実装済み
  - Unboundではharden-referral-pathで対策済
  - Nominum Vantioではいろいろな対策を実装済
  - BIND 9.10では、ISC独自のDNS Cookieを実装
  - Google Public DNS: nonce prefix, 多数のIPアドレス使用

Copyright © 2014 Japan Registry Services Co., Ltd.

まとめ

- 委任情報を注入する攻撃は容易である
  - ほとんどのドメイン名を攻略できる
  - しかし、足がつく
- ソースポートランダマイゼーションで攻撃の効果を弱くできる
  - 放置されているフルリゾルバの存在
  - よく管理されていると被害をうけにくい
- 攻撃実績を聞いたことがない
  - 多くのひとは対応済だと考えている
  - 費用対効果を考えた対策が重要である

Copyright © 2014 Japan Registry Services Co., Ltd.