Chrome はなぜTCPクエリを出したのか

草場 健 @ IIJ

2023/06/23 DNS Summer Day 2023

自己紹介

- 名前: 草場 健 (くさば たけし)
- 所属: 株式会社インターネットイニシアティブ
- 仕事: 自社製ルータ(SEIL)のファームウェア開発
  - DNS周りやレイヤー低めのところなど

背景

- ChromeがTCPでDNSクエリを出すようになった
- 自社製ルータでの影響確認の指示があった
- そもそもの原因を調べて社内で共有したところ、
  一定の人数にウケたので追加調査して今回発表することに
- 注意:
  - 発表者はChromeの開発者・関係者ではありません
  - 内容は自分でソースコードを読んだり、手元で検証したものです

再現させよう

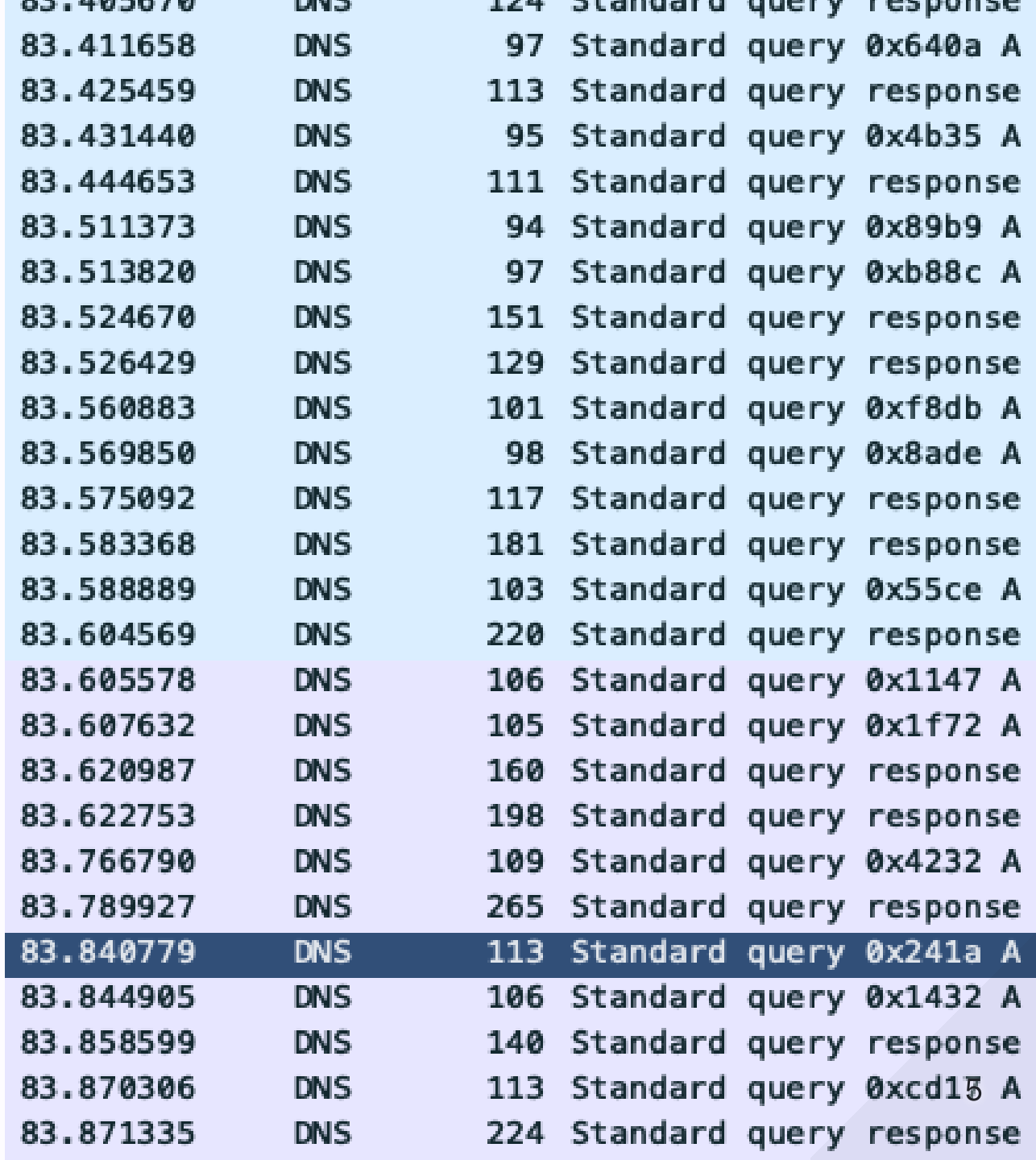
- Macでブラウジングしても再現しない🤔
- Windowsでブラウジングをしても再現しない🤔
- Windowsで**built-in resolverを使うと再現**💡
  - 起動直後はUDPクエリを出す
  - しばらくブラウジングしているとTCPクエリになる

再現させよう

あるタイミングを境にTCPでクエリを出している様子

青: UDP
紫: TCP

- Chrome 112.0.5615.140
- Windows 10 21H2 19044.2728



なぜTCP？

- 愚直にソースコードをあたってみる
- クエリを出してそうな部分を見ると、`low_entropy()` という関数(中身はフラグ)でTCP/UDPが分かれている

```
if (session_->udp_tracker()->low_entropy()) {
    result = MakeTcpAttempt(server_index, std::move(query));
    RecordAttemptUma(DnsAttemptType::kTcpLowEntropy);
} else {
    result = MakeUdpAttempt(server_index, std::move(query));
    RecordAttemptUma(DnsAttemptType::kUdp);
}
```

https://github.com/chromium/chromium/blob/112.0.5615.178/net/dns/dns_transaction.cc#L1323-L1329

なぜTCP？

- built-in resolverはエントロピーが足りなさそうな挙動を観測すると、`low_entropy()` フラグが立つ (1回フラグが立つと戻らない)
  - 条件の1つ「ポート番号が直近256回の内2つと重複」にあたってそう
    - (※) resolver はポート番号をランダムにして毒入れされないようにしたい
  - 観測時のキャプチャから抜粋↓

```
09:40:33.729087 IP XXX.49664 > YYY.domain: 47528+ A? ...
09:40:33.741029 IP YYY.domain > XXX.49664: 47528 2/0/0 A ...
09:40:34.639129 IP XXX.49664 > YYY.domain: 21966+ A? ...
09:40:34.654809 IP YYY.domain > XXX.49664: 21966 4/0/0 ...
（この後TCPに切り替わる）
```

  - おそらくこの後にsrc port 49664のパケットを出そうとして発火

なぜTCP？

挙動と整合性のある説明ができる

- 最初はUDPを使う ⇔ 起動直後はエントロピー不足判定されてない
- しばらくするとTCPになる ⇔ エントロピー不足判定された

さらに

- built-in resolverはChrome 109からデフォルト利用に
  - Chrome 109のリリース日は2023/01/10

しばらくして...

各地で報告がされ始める

- Chromiumでのissue
  - 1413620 - Huge amount of "ERR_NAME_NOT_RESOLVED" error
  - 目をつけた場所はあってた
  - ポート被りはWindowsのポート番号範囲が狭いのが原因と推定
  - Mac, Linuxではポート割当方法が違うのでWin固有の問題とのこと
- DNS-OARCのdns-operations ML
  - Increase in DNS over TCP from Chrome Browser on Windows 11
  - ポート被りはソケットキャッシュ(?)が原因と推定

→ 原因を確かめよう！

試す

ざっくり Chrome と同じ方法でソケットを用意して、重複回数をカウントしてみる

```
for (i=0; i<1024; i++) {
    DWORD r = 1;
    sock = WSASocket(AF_INET, SOCK_DGRAM, IPPROTO_UDP, NULL, 0, WSA_FLAG_OVERLAPPED);
    setsockopt(sock, SOL_SOCKET, SO_RANDOMIZE_PORT, (const char*)&r, sizeof(r));
    connect(sock, (struct sockaddr *)&sa, sizeof(sa));
    getsockname(sock, (struct sockaddr *)&sa, &sa_len);

    // htons(sa.sin_port) がポート番号なのでここで集計する

    closesocket(sock);
}
```

コード概要

試す

少し多めに1024回繰り返して、ソースポート番号の重複を見てみる

結果...

試す

少し多めに1024回繰り返して、ソースポート番号の重複を見てみる

重複	ポート数
13	1 (?!)
3	1
2	30
1	948

TCPクエリの再現時と同環境で実施

少なくとも、完全にランダムでは無さそう...

試す

複数のWindows環境で試してみるも、原因は分からず
4, 5はOSは同じなのでハード要因の可能性もある

#	OK?	エディション	バージョン	ビルド番号
1	✅	11 Home	22H2	22621.1702
2	❌	11 Pro	22H2	22621.1555
3	✅	10 Pro	22H2	19045.2006
4	✅	10 Pro	21H2	19044.2728
5	❌	10 Pro	21H2	19044.2728

その後

- エントロピー不足の判定基準を厳しくする変更が入った
  - ポート番号が直近256回のうち2回被る -> 3回被る
- 2023/05にパッチ適用済みのChromeがリリースされる
  - だが不具合が発生していた環境では引き続き再現...
- 今後どうなるのでしょうか

おまけ: MacやLinuxでは？

- 結論: 実装が違う
- WindowsではOS側に任せている
- MacやLinuxではChrome側が生成した乱数をソースポートに使う
  - 前からbuilt-in resolverを使っていたが、発生しなかったのはこれ
- Windowsで同じことをしようとするとファイアウォールの警告が出るらしく、今の形式になったとのこと

まとめ

- Chromeが突然TCPでクエリを出すようになった
- おそらくWindowsのポート割当アルゴリズムとChromeのセキュリティ機構の相性問題
- To be continued...

ご清聴ありがとうございました

タイムライン

時期	出来事
2022/11/02	Windows版でbuilt-in resolverがデフォルト有効に
2023/01/10	Chrome 109リリース
2023/01中旬	SNSでユーザから報告が出る
2023/02/07	bugs.chromium.orgで報告される
2023/03/29	対応パッチが入る
2023/05/08	パッチ適用済みのChromeがリリース