Internet インターネット 10分講座

DNSキャッシュポイズニング

今回の10分講座は、最近になって新たな攻撃方法が発見され、対応の緊急性が高まったDNSキャッシュポイズニングについて解説します。

■DNSの問い合わせの流れ

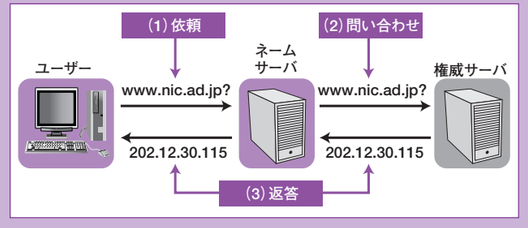
まずはじめに、DNSではクライアントがどのようにドメイン名の情報を得るのか、その流れについて説明します（図1）。

（1）エンドユーザーのPCなどのDNSを利用するクライアントから、問い合わせを行うネームサーバに対し、問い合わせを依頼します。

（2）依頼を受けたネームサーバは、問い合わせ内容を元に、ルートサーバから委任をたどりながら順に問い合わせを行い、目的のドメイン名情報を持つ権威サーバから結果を取得します。

（3）依頼を受けたネームサーバは、問い合わせの結果をクライアントへ返答します。

図1：DNS 問い合わせ



■DNSのキャッシュ

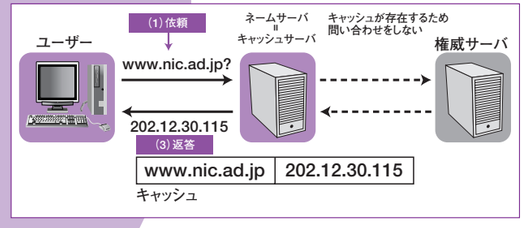
問い合わせを処理するネームサーバは、処理の途中で得たドメイン名の情報を一時的にローカルに保存することができます。この処理をキャッシングといい、一時保存した情報をキャッシュといいます。

過去に行った内容と同じ問い合わせをする場合は、他のネームサーバへ問い合わせることなく、キャッシュとして保持している情報を利用してクライアントに返答します。データをキャッシュに保存しておく期間は、権威サーバがそれぞれのデータに対して指定することができ、これをTTL（time to live）と呼びます（図2）。

このキャッシュという仕組みによって、他のサーバへの問い合わせ回数の減少、DNSの負荷やネットワーク帯域の軽減、問い合わせにかかる時間の短縮といった効果が得られます。

クライアントから問い合わせの依頼を受けるネームサーバは、このキャッシュの仕組みを実装していることが多いため、「キャッシュサーバ」とも呼ばれています。

図2：キャッシュ



■DNSキャッシュポイズニング

DNSでは、前述の通りキャッシュという仕組みによって、負荷の軽減や高速化を図っています。この機能を悪用し、キャッシュサーバに偽のDNS情報をキャッシュとして蓄積させる「DNSキャッシュポイズニング」と呼ばれる攻撃があります。

DNSキャッシュポイズニングによる影響はさまざまですが、攻撃を受けたキャッシュサーバを利用するユーザーに対して、以下のような影響を与えることが可能です。

（1）ホスト名とIPアドレスの対応を変更し有害サイトへ誘導する
（2）Web、メールの内容を盗聴する、改ざんする
（3）spamを送信する
（4）DNSを使用不能にして、各種サービスやアプリケーションを動作不能にする（DoS）

DNSキャッシュポイズニングには、DNSサーバソフトウェアの不具合や設定間違いなどを狙った攻撃がありますが、以下では、DNSプロトコルの弱点をつく攻撃方法について説明します。

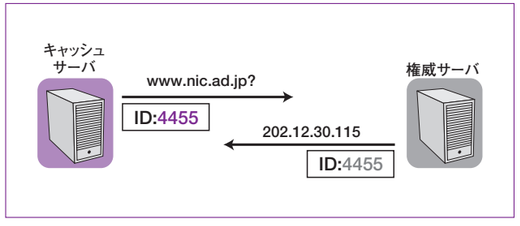
■DNSメッセージの偽装と従来の攻撃手法

DNSでは、DNSメッセージ中に“ID”という16ビットの識別子が用意されています。これは、応答メッセージを受信したときに、それがどの問い合わせメッセージに対するものかを判断するために使用されます。

キャッシュサーバは、問い合わせメッセージ中にIDを指定して送信し、あとでDNSメッセージを受信したときにそのメッセージ中のIDを調べ、問い合わせたときのIDと一致するものを、対応する応答メッセージとして処理します。

問い合わせのときに指定したIDと、応答パケットに含まれるIDが一致しない場合は、不正なDNSメッセージとして破棄します（図3）。

図3： DNSメッセージのIDによる識別



ただし、問い合わせたときに指定したIDと、受け取ったパケット内のIDが一致していた場合、キャッシュサーバは、受け取ったパケットが正しい権威サーバから送られてきたものと判断してしまい、偽装されたDNSメッセージであっても問い合わせの結果として処理してしまいます。

DNSの問い合わせや回答は、主にUDPを用いて通信が行われます。UDPはTCPよりも通信にかかるコストが低い反面、通信パケットの偽装が比較的容易なため、パケット中のIPアドレス、ポート番号、IDを細工することで、DNSメッセージを偽装することが可能です。

特に、IDの取る値が16ビット（=65536通り）しかないので、推測や総当たりで一致させることは不可能ではないことが以前から指摘されており、そのため、このDNSプロトコル上の脆弱性を利用したキャッシュポイズニングの攻撃手法は古くから知られていました（図4）。

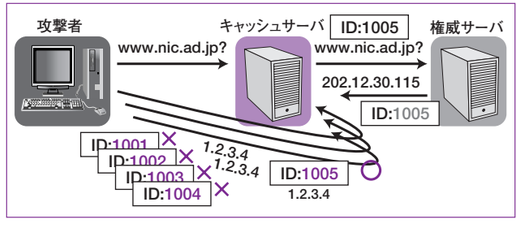
1. 攻撃者は、偽の情報を送り込みたいドメイン名について、ターゲットとなるキャッシュサーバに問い合わせを送る

2. 問い合わせを受けたキャッシュサーバは、外部の権威サーバに問い合わせる

3. 攻撃者は、権威サーバから正しい応答が返ってくる前に、偽の応答パケットをキャッシュサーバに送り込む

4. キャッシュサーバが 2.で送った問い合わせメッセージのIDと、攻撃者が3.で送った偽のメッセージのIDが一致した場合、攻撃成功

図4： 従来の攻撃手法



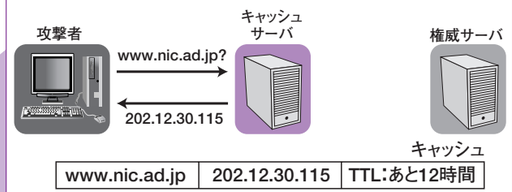
ただし、キャッシュポイズニング攻撃は、キャッシュサーバの問い合わせの応答パケットを偽装する必要があります。

また、キャッシュの有効な間（TTL）、キャッシュサーバは外部の権威サーバへの問い合わせを行いません。つまり、偽装応答パケットを送り込むチャンスは、有効期限が切れる（TTLの長さ）ごとに1回となるため、IDを一致させること自体は困難でなかったとしても、TTLを十分に長く設定しておけば、キャッシュポイズニングの成功率は低く抑えることができると考えられていました（表1、図5）。

表1：従来の攻撃手法で必要な条件

1. 攻撃者は、キャッシュサーバが問い合わせで使用したIDと偽装応答パケットのIDを一致させる
2. 攻撃者は権威サーバからの応答よりも早く偽の応答パケットを送り込む
3. キャッシュに存在しない、あるいはキャッシュの有効期限が切れている

図5：TTLによる成功率の減少



■新しい攻撃方法（Kaminsky attack）

しかし、2008年8月、セキュリティ研究者のDan Kaminsky氏によって、新たなキャッシュポイズニング攻撃の方法が発表されました。前述の通り、これまではTTLが十分に長ければ、成功率を低くできると考えられてきましたが、Kaminsky氏による攻撃手法では、TTLの長さに関係なく攻撃することが可能であるため、その前提が崩れました。

Kaminsky氏の発見した攻撃手法は以下の通りです。

（1）攻撃者はターゲットのキャッシュサーバに対し、乗っ取りたいドメイン名と同じドメイン内で、存在しないドメイン名を問い合わせる。
例えば www.example.jp のドメイン名を乗っ取る場合、〈ランダム文字列〉.example.jp を問い合わせる。
ex.“12pqr3s4.example.jp”

（2）問い合わせを依頼されたキャッシュサーバは、キャッシュにないため権威サーバに問い合わせる

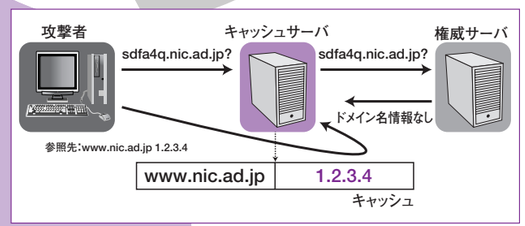
（3）攻撃者は偽の応答として参照先を示す内容のパケットを、ターゲットのキャッシュサーバに送り込む

たとえば
「そのドメイン名については、他の権威サーバに問い合わせよ。そのサーバ名は www.example.jp、IPアドレスは aa.bb.cc.dd」
（aa.bb.cc.ddは攻撃者が用意した偽の権威サーバのIPアドレス）

この攻撃方法は、ランダムなドメイン名をキャッシュサーバに問い合わせることで、強制的に外部に問い合わせを行わせられること、また、IDの不一致などで攻撃に失敗しても、ドメイン名を変えてすぐに再チャレンジできることから、これまで考えられてきた方法よりも効率的にキャッシュポイズニング攻撃が行えます（図6）。

この手法は Kaminsky氏の名前をとり “Kaminsky attack” と呼ばれています。

図6：Kaminsky attack



■Kaminsky attack への対処

（1）Source port randomization

従来の攻撃手法においても、Kaminsky attackによる手法においても、その前提として、DNSでの応答パケットが偽装されているかどうかの判断がIDによって行われており、そのIDが16ビットの値しか取れず総当たり可能となっていることを、脆弱な点として狙っています。従って、対応策としては、例えばIDのビット長を32ビットや64ビットなどに長くすれば、IDの推測は困難になり、攻撃することも難しくなります。しかし、ビット長を長くするには、DNSプロトコルそのものを変更しなければならないため、その実現はほぼ不可能と言えます。

そのため、偽装応答パケットの生成を困難にする別の方法として、Source port randomization という手法が考えられました。これは、キャッシュサーバから権威サーバへ問い合わせるときに使用するUDPポート番号を、固定あるいは狭い範囲で使用するのではなく、広範囲な番号からランダムに選択して通信に使用することによって、応答パケットの偽装を難しくさせる方法です。偽装の難易度は、ポート番号の利用範囲に比例して難しくなります（図7）。

この Source port randomization は、各DNSベンダーによりパッチがリリースされています※。

ただし、このSource port randomizationも、キャッシュサーバの設定、NAT、ファイアウォール装置などによって、外部から見た問い合わせポートが固定・限定されることがあり、効果が減少する場合がありますので注意が必要です。

図7：Souce port randomization によって生じる推測の必要なパターン数

	ポート	ID	組み合わせ数
固定（1ポート）	1	× 65536 =	約6万5000通り
乱数（100ポート）	100	× 65536 =	約650万通り
乱数（32000ポート）	32000	× 65536 =	約20億通り

※Vulnerability Note VU#800113
- Multiple DNS implementations vulnerable to cache poisoning
http://www.kb.cert.org/vuls/id/800113

（2）問い合わせDNSクライアントの限定やパケットフィルタ

キャッシュポイズニング攻撃は、攻撃者がDNS応答パケットを偽装してキャッシュサーバに送り込むことで行うことは既に述べました。そのため、攻撃者が自由に攻撃できないようにすることが必要です。つまり、キャッシュサーバに問い合わせ可能なクライアントを限定すること、ソースアドレスの偽装されたパケットを遮断することなどで、リスクを軽減することができます。さらに、主に攻撃方法は、IDを総当たりで推測するというものであるため、異常なDNSパケット（問い合わせに使用していないIDの応答があったり、権威サーバからの応答が問い合わせた回数以上に大量にあるなど）が観測されます。これらを検知し防御することで、キャッシュサーバを保護することも可能です。

■ その他のKaminsky attackへの対応策について

（1）DNSSEC

根本的な解決策としては、DNSSECがあげられます。

DNSSECでは、権威サーバによって応答に電子署名が行われ、キャッシュサーバがその署名を検証することで、応答が偽装・改ざんされているかどうかを確認できるようになります。そのため、キャッシュポイズニング攻撃に対する防御はほぼ完全になります。

しかし、DNSSECを利用するには、権威サーバ・キャッシュサーバのDNSSEC対応、電子署名に必要な鍵の管理や配布方法の確立、ルートやTLDの署名が必要なことなど、さまざまな課題があります。

従って、DNSSECが普及するにあたっては今しばらく時間が必要であり、早急な対応が求められる現状では時間が不足しています。

（2）SSL（Secure Socket Layer）

偽のドメイン名情報により別サイトに誘導されたとしても、通信の接続時にSSLによる認証を行っていれば、通信相手が偽のホストかどうか検知できます。しかし、DNSキャッシュポイズニングによる影響はWebだけではなく、電子メールやFTPなど多岐にわたるため、SSLは部分的な防御にしかなりません。また、Webに限ったとしても、偽のサーバに接続したときに出るWebブラウザの警告を無視してしまうユーザーに対しては、残念ながらSSLは有効な手段になりません。

■ 参考

キャッシュサーバが、IDおよびソースポートについて十分な乱数性を持っているかどうか確認することのできるサービスを、DNS-OARCが提供しています。

Web-based DNS Randomness Test
https://www.dns-oarc.net/oarc/services/dnsentropy

また、キャッシュサーバが、問い合わせ可能なクライアントを無制限にしていないかどうか確認できるサービスを、IANAが提供しています。

Cross-Pollination Check
http://recursive.iana.org/

（JPNIC 技術部 小山祐司）