IDEON-retreat レジュメ: 担当 doi 2006-04-03

V. Pappas, D. Massey, A. Terzis, L. Zhang, “A Comparative Study of the DNS Design with DHT-Based Alternatives” In the Proceedings of IEEE INFOCOM’06, Apr. 2006.

1 概要

DDNS や CoDoNS と呼ばれる DHT ベースの DNS 互換サービスが提案されている。本論文では、これを DNS と比較し、それぞれの利点と欠点を評価した。DNS と比べ、DHT ベースのシステムは、攻撃者がノードを選択的にダウンさせられるという状況における安定性 (resilience) が高い。一方、オーバーレイ上の経路選択のメカニズムの都合で、ランダムなノード故障に対しては DHT は必ずしも有利であるとは言えない (但し、静的な環境において)。

一方、名前キャッシュの有効性を決定づける仕組みが DNS と DHT で全く異なることも示している。DNS ではローカルな名前問い合わせの出現頻度と名前空間木の分岐数 (厳密にはそのゾーンを refer する問い合わせの出現頻度) によりキャッシュの有効性が決定づけられる。これに対して、DHT ではグローバルな名前問い合わせの出現頻度とシステムのノード数によりキャッシュの有効性が決定づけられる。

ここでは、最新の DNS と最新の DHT の比較、というよりは、それらのノード間のリンクの構成の方法 (構造) が決定づけるシステムの安定性とキャッシュの有効性を評価している。また、比較のためのアルゴリズムには Chord が採用されている。

2 定性的な議論

この論文で評価しているのは、安定性と性能である。ノード間の構造は安定性と性能に次のようにかかわっている。

ノード間の構造による効果、という視点に立つと、ノード間の経路の冗長性とキャッシュの効果による経路の短縮がそれぞれ安定性と性能にかかわる効果である。

経路の冗長性では DNS が有利である。ここで、キャッシュが一切存在しないと仮定して DNS と DHT を比較する。

DNS では、合計 i 段の問い合わせ (名前空間木の深さが i) において、それぞれの段 i を担うノード数の積が、ルートから目的の名前へ辿りつく経路の数となる。つまり、組み合わせのうち 1 つでも活きていれば問い合わせは成功する。一方、DHT で仮に i ホップの問い合わせを行うとしても、途中で経路の袋小路に迷い込む可能性がある。例えば、経路の途中に選択したノードの隣接ノードが全てダウンしていた場合は、経路を戻り異なる経路を辿れば問い合わせが成功する可能性があるにもかかわらず、問い合わせが失敗してしまう。

また、キャッシュの効果は DNS では名前空間木に対して利くため、有名な名前の親については経路短縮効果が大きくなる (NS referral)。一方、DHT では名前は全てハッシュ関数にかけるため、個々の名前単位でキャッシュの効果が発生する。その結果、全世界的に問い合わせ頻度の高い名前については、経路の途中でキャッシュに当たる可能性が高くなる。ただし、DHT を構成するノード数が増えれば増えるほど権威を持つノードまでの経路長が長くなり、キャッシュに当たる可能性が増えたとしても相対的には問い合わせの性能は劣化する。

3 評価手法

評価のメトリックには、次の 3 つを用いている。これらのメトリックからわかる通り、この研究では L7 の構造のみを検討している。

Data Failure Rate: 権威を持つノードの故障による問い合わせ失敗の比率

Path Failure Rate: 権威を持つノードが少なくとも一つ利用可能であるにもかかわらず、経路上のノードが利用できないための問い合わせ失敗の比率

Path Lengths: 成功した問い合わせにおいて、問い合わせを行うのに費やしたホップ数

また、評価の前提となる問い合わせの性質を決めるために、著者らが独自に収集した DNS のトラフィッ

クを利用している。このトラフィックはDNSの性質解析と、DNS/DHTのシミュレーション環境の入力として使われる。ただし、DHTにおいてはネットワーク全体での問い合わせ頻度がキャッシュの効果に影響する。この影響を見積るため、評価のために着目するクライアントからはトラフィックから得られる問い合わせの列をそのまま発行し、その他のクライアントにはトラフィックから得られる問い合わせの頻度を保存したランダムな問い合わせを発行する。ただし、ここで名前ごとの問い合わせの頻度が全世界で共通であるという仮定を置いている。

安定性への影響を評価するために、独自開発のシミュレータを用いている。シミュレータでは、キャッシュ機能を無効にした評価と有効にした評価を比較することで、キャッシュの効果と構造自体が持つ安定性への効果を分離している。DNSではトレースから得られる名前サーバをノードとして設定し、DHTでは経路の底(base of the ring)が$2, 4, 8$となる1024〜65536ノードのリングを構築し、全ての名前をinsertした。それぞれにおいて、ランダムあるいは任意のノードを無効として、トレースと同じ入力があった場合の問い合わせの成功(および経路長)と失敗(およびその理由)を記録した。

4 解析モデル

このモデルとシミュレーショントレースとが符合するか検証している。とりあえず省略。

5 評価結果

5.1 利用可能性

ランダムな故障に対して、DNSにおいてはpath failure rateのほうがdata failure rateよりも低いのに対して、DHTにおいては逆の結果となった(Fig. 2および3)。DNSにおいてはより上位のゾーンは多数のレプリカを持っており、path failureは相対的におこりにくいと考えられる。一方DHTにおいては、経路長の平均が相対的に長く、また利用可能な経路が全て活用されていないという理由があると考えられる。

なお、これらの結果はノード数とはそれほど大きく影響しないとしている。DNSについてはノード数が変化しても、あるゾーンを構成するノードが失敗する可能性は変わらない。一方DHTの場合、ノード数が大きくなると経路長も長くなるため失敗の可能性が上がる効果があるが、この効果は経路の選択肢が増えることにより相殺される。

また、Fig.6および7は、ノードのランダム故障に対して、成功した問い合わせの経路長の変化の割合を示している。[1]DNSでは、故障率10%においては約30%の問い合わせが1つ余分な問い合わせを行っている。DHTは、DNSに比べて経路長の増分が大きくなる傾向がある。これはDHTの経路長の平均がDNSに比べて長いためである。実際のDHTではそれぞれの経路を定期的にモニタし、故障したノードを経路から取り除く。その場合DHTと比較できる程度の経路長の増分になるという[文献12]。

Table IIIはDNSとDHTの問い合わせ失敗の比率を示している(DHTは8192ノードの10%が問い合わせを発行する条件)。DNSにおいては、Cacheの効果はPath Failureの低減としてあらわれる。仮に90%のノードが落ちていたとしても、Path Failureは17.59%であるという(!)。しかし、Data Failureはほとんど改善しない。

一方、DHTにおいてはCacheを有効にすることで、Path FailureとData Failureの両方が少しづつ改善する。Data FailureおよびPath Failureの改善効果は偶然経路上にキャッシュがある場合のみに発生する。特に、Path Failureの改善効果は、問い合わせ経路上に対象となるデータのキャッシュが存在し、かつキャッシュの存在するノードとそのデータに権限を持つノードとの間のPathが不良な場合のみ発生する。

また、悪意を持った攻撃に対する評価は、100、500、1000ノードを恣意的に選択し、3〜24時間停止した場合のPath Failureを評価している[2]。DNSではツリーの上位のノードを選択して攻撃されたとし、停止させている。停止ノード数が100ノード程度の場合はTLDやSLDのNS cacheによりFailureの比率がかなり低く抑えられる反面、500〜1000と増加するにつれ大幅にfailure rateが増大する。一方、DHT

[1] doi:どうやら問い合わせタイムアウトはコストとして算定されていない模様

[2] 停止中のPath Failureのみを評価している？

では連続した領域のノードを停止させているが、停止する台数の増加による影響は小さい。

5.2 キャッシュと性能

(時間切れにつき御免)

DNS: Fig. 9: サブドメインを持つ数とキャッシュヒット率の散布図。Fig. 10: NS referral の到着頻度とキャッシュヒット率の散布図。問い合わせ応答に含まれる NS Referral によりキャッシュが refresh できるので、頻繁に利用される名前を含むゾーンほどキャッシュの有効性が上がる。

DHT: Fig. 11: 問い合わせホップ数の分布 (クライアントの比率変化、ノード数固定)。Fig. 12: 問い合わせホップ数の分布 (クライアントの比率固定、ノード数変化)。Fig. 13: 問い合わせホップ数の分布 (クライアントの数固定、ノード数変化)。少ないレコードに対する問い合わせをたくさん処理するのであれば普通の cache /w たくさんのリゾルバクライアント in DHT で足りるが、逆にたくさんのレコードを少ないリゾルバがアクセスする環境では、proactive cache などを使わないとキャッシュの効果がでづらい。Fig.13 2^20nodes はキャッシュの効果がほとんどない分布。