輪講資料 04/11/4 @ IDEON Fall Retreat
Emil Sit and Robert Morris, “Security Considerations for Peer-to-Peer Distributed Hash Tables”.
資料作成: 門林 雄基（奈良先端科学技術大学院大学）

## Abstract

DHT に基づく大規模 P2P システムには、どのようなセキュリティ問題が内在しているかを考察する。問題、例を挙げ、それらを検知し防ぐための設計原理を提案する。（正確かつ網羅的な議論ではない。読んでインスピレーションを得るような論文。）

## 1. Introduction

DHT への攻撃として、壊れたデータを返すものがある。データの正当性、信憑性は暗号学的手法で実現できる。

この論文ではシステムの liveness への脅威となるような攻撃に焦点をあてる。例えば参加者が該当データを見つけられない等。

Table 1: この論文での議論から導き出した一般的な設計原理。

## 3. Adversary Model

この論文で考える攻撃者は DHT の参加者で、プロトコルに正しく従わないノードである。正しいノードをミスリードし、誤った情報を与えようとする。

ノードは任意のパケットを生成できるが、自分あてのパケットしか読み取ることができないものとする。つまり通信を盗聴したり中間奪取することはできない。この状況では IP アドレスは弱いノード ID として使うことができる。つまり 3-way handshake のようなことをやると、お互いそのアドレスを所有していることが確認できる。

また悪意のあるノードは共謀できるものとする。

以下では、悪意のあるノードがこれらの能力を用いてシステムをないがしろにする方法についてみていく。

## 4. 攻撃と防御

routing, data storage system それぞれへの攻撃、および一般的な考察について順に述べる。

攻撃の防御では、まず検出することが重要。しかし検出したあと何をすべきかは明らかではない。悪意がある場合だけでなく、騙されていることが分からない場合もあるかもしれない。

ここでの議論では、まず検出し、次に可能であれば一貫性を欠く情報を正すことを考える。検証可能性 (verifiability) を確保することがすべての検出手法の要となる。

## 4.1 経路攻撃

DHT では、ルーティングの正しさが重要。攻撃者がこの点を突いてくる可能性は大きい。この手の攻撃は、システムにおいて検証可能な不変特性 (verifiable system invariants) を定義すれば検出できる。これが実現できない場合、システムは回復機構を持つ必要がある。

### Incorrect Lookup Routing

悪意のあるノードは lookup を誤った、あるいは存在しないノードに転送できる。

幸い、明らかに間違った転送は用意に検出できる。各ホップで、目的値に近づいているはず。問い合せを発行したノードはこの点を確かめ、攻撃を見つけたならば、ひとつ前のホップに戻り、代替手段（別経路など）を頼めばよい。

この検査をおこなうためには問い合せノードに進展が見えていないといけない。例えば CAN では、ルーティングを最適化すると、この条件を満たさない。

また悪意のあるノードはでたらめなノードを key に対応すると言い放つかもしれない。この問題への解決策は 2 段階からなる。

まず問い合せを発行したノードは、目的ノードが実際にその問い合わせの到達点 (query endpoint) であることに合意しているか確かめる必要がある。Chord では、predecessor が query endpoint のアドレスを返す。predecessor に悪意がある場合、到達点 $S$ より後ろの $S'$ を教えるかもしれない。この場合、問い合せノードは悪意を検出できない。しかし $S'$ が善良であれば、その key を自身がもつべきでないことを検出しエラーを発生させることはできる。

次にシステムは、key をノードに、検証可能な形で割り当てるべきだ。例えば CAN では任意のノードが自身の ID を決めることができる。これでは他のノードは検証できない。Chord では IP アドレスとポートをハッシュ関数にかけることでこれを防ごうとしている。

システムで公開鍵を用いて長期的な ID を与えることもできる。公開鍵は検証可能性を改善するだろう。例えばアドレスと公開鍵の証明書。

### Incorrect Routing Updates

誤った経路情報。これも正しさが検証可能であればなんとかなる。（あまり正確でない議論。）

近接性などより複数のサーバを選べるシステムでは、攻撃者がサーバ選択を悪用できる危険性がある。例えば匿名ネットワークで近接性を利用し共謀するサーバを選択させ、匿名性を失わせるなど。（例のみにたよった正確でない議論。）

### Partition

ブートストラップ時に間違ったネットワークに入る危険性について。善意のネットワークとまったく同じプロトコルを用いる、悪意のあるノードで構成されたネットワークを考える。新規ノードが誤って悪意のネットワークに加入するよう工夫する。善意のネットワークと同様のコンテンツを持っていたり、一部のノードが善意・悪意の両方のネットワークに加入するなど。

これはサービス妨害攻撃や、クライアントの挙動を解析するときに使える。

これはブートストラップ問題。信頼できる情報源からブートストラップすべき。システム以外の情報源。(phishing などを見ると、それだけで充分なのか疑問。）公開鍵。（問題の先送りでしかな

いことが、どうやら分かっていないらしい。）

断片的アイディア：ほかのノードに random query を頼んで、その結果と自分の random query の結果を照らし合わせる。（役に立つのか疑問。）

## 4.2 Storage and Retrieval Attacks

## 4.3 その他の攻撃

### 一貫性のないふるまい

### 標的となるノードの過負荷

### 迅速な加入と離脱

### 頼まれもしないメッセージ