Sicherheit in Peer to Peer Netzwerken

Patrick Hochscheidt

1. Dezember 2006

1 Einleitung

Da Peer to Peer Systeme bisher weniger zum Einsatz kommen als Client-Server-Systeme basieren Angriffe und Verteidigungen für Peer to Peer Systemen meist auf theoretischem Material und sind daher kaum getestet oder simuliert worden. Deshalb gibt es noch eine große Anzahl an Ungereimtheiten bei den Verteidigungen und Angriffen die noch zu erforschen sind. In dieser Arbeit wird auf verschiedene Angriffe eingegangen und mögliche Verteidigungsmaßnahmen vorgestellt.

2 Sybil Angriff

Der Begriff „Sybil"-Angriff ist vom Titel eines Buches abgeleitet. Das Buch Sybil wurde von der Psychologin Flora Rethas Schreiber geschrieben und 1978 veröffentlicht [8]. Sie beschreibt in dem Buch den psychologischen Fall einer jungen Amerikanerin die 16 Persönlichkeiten aufweist. Hierbei ist jede Persönlichkeit eigenständig, unter anderem eine amerikanische Kellnerin und ein französisches Kind.
In einem Peer to Peer System beschäftigt sich der Sybil-Angriff damit, multiple Identitäten von einem Rechner ausgehend zu etablieren [8]. Diese neuen falschen Identitäten können z.B. Suchanfragen fehlleiten oder Mehrheitsabstimmungen manipulieren. In einem Peer to Peer Netz mit 100.000 Knoten und hiervon 100 bösartigen Knoten wird das Routing bereits um 35 Prozent verlangsamt [7]. Das wirklich gefährliche am Sybil Angriff ist, dass er auch als Vorbereitung für andere Angriffe wie z.B dem Eclipse Angriff genutzt werden kann vg.Kapitel3.
Um falsche Identitäten zu verhindern gilt es die Frage der Verifizierung neuer Identitäten zu klären. Dafür gibt es nur zwei Möglichkeiten, entweder erlaubt man nur einer zentralen Instanz neue Identitäten zu verifizieren oder man lässt bereits bestehende Identitäten neue Identitäten verifizieren.
Bei der Verifizierung einer Identität durch andere Identitäten ist es schwierig, die Menge neuer Identitäten zu begrenzen. Eine einfache Möglichkeit ist die Aufgabenstellung von cryptographischen Rätseln. Um das Rätsel zu lösen muss z.B. eine aufwendige mathematische Hashfunktion gelöst werden. Ältere Rechner könnten aber damit überfordert sein da dies sehr rechenintensiv ist. Außerdem können Angreifer durch die Übernahme von Rechnern an theoretisch unbegrenzte Ressourcen kommen. Man kann durch so eine Strategie nur die Anzahl der neuen Identitäten verlangsamen. Hierbei ist problematisch, dass bösartige Identitäten neue, eigene Identitäten verifizieren können. Dadurch kann eine unbegrenzte Anzahl an neue Identitäten erschaffen werden vg.Abb.1.

In [1] wird bewiesen, dass eine eindeutige Verifizierung von neuen Identitäten ohne eine zentrale Instanz nicht möglich ist. Es wird gezeigt, das jede bösartige Identität unendlich viele neue Identitäten, in einem System ohne zentrale Instanz erschaffen kann. Das Problem bei einer zentralen Instanz ist, dass diese für einen Denial of Service Angriff anfällig ist. Außerdem ist ein Netz mit einer zentralen

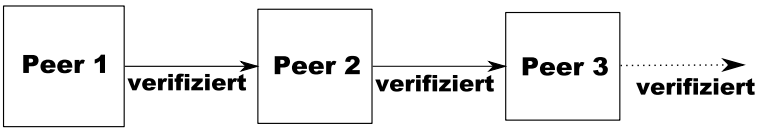

Abbildung 1: Abb1. Die Verifizierung durch andere Identitäten

Instanz gefährdet, wenn diese ausfällt, übernommen oder überlastet wird. Ein Überlasten oder ein Ausfall kann zum Stillstand eines gesamten Netzes führen.

3 Eclipse Angriff

Der Eclipse Angriff steht für das „Dunkel-Schalten“ von Knoten, das bedeutet, dass diese „dunklen“ Knoten nur noch an bösartige Knoten weiterleiten können [2]. Da dunkel geschaltete Knoten nur noch an bösartige Knoten weiterleiten und diese entscheiden können was sie weiterleiten und was nicht, wird systematisch erst Knoten und dann Teilnetze ausgesperrt. Mit dem Ziel das Overlay unter die Kontrolle der Angreifer zu bringen.
Das dunkel Schalten von Knoten wird durch falsche Routing-Tabellen erreicht. Diese falschen Routing-Einträge werden von bösen Knoten erzeugt. Die bösen Knoten nutzen hierfür die Aktualisierungsfunktionen des Systems, böse Knoten geben keine, falsche oder bösartige Adressen zurück. Je mehr falsche Einträge ein Knoten hat, desto „dunkler“ wird er, d.h. desto weniger richtige Einträge hat er an die er weiterleiten kann.

3.1 Anfälligkeit für Eclipse Angriff

Unstrukturierte Netze sind am anfälligsten für einen Eclipse Angriff, da die neue Knotenfindung durch Flut oder zufällige Wegfindung geschieht. Dadurch ist die Wahrscheinlichkeit hoch, auf bösartige Knoten zu stoßen, die falsche oder bösartige Adressen zurückgeben. Außerdem legen unstrukturierte Netze keinen Wert auf die Menge ihrer Einträge in ihrer Nachbarschaftsmengen. Hierbei ist der Anzahl an falschen Einträgen keine Grenzen gesetzt. Daher ist es für einen Angreifer leicht, diese Knoten zu übernehmen.

Strukturierte Netze sind wesentlich widerstandsfähiger gegen einen Eclipse Angriff da sie die Menge ihrer Nachbarschaftsmenge begrenzen. Durch diese Begrenzung sinkt die Wahrscheinlickeit auf einen bösartigen Knoten in der Menge zu stoßen. Außerdem besitzt jeder Peer eine eindeutige Identifikation die schwierig zu fälschen ist, da diese von einer zentralen Instanz oder von Protokollspezifikationen abhängt.

3.2 Degree Bounding

Eine Verteidigung gegen einen Eclipse Angriff ist das *Degree Bounding*. Beim Degree Bounding wird angenommen, dass jeder Knoten einen *indegree* und *outdegree* hat vg.Abb.2. Der indegree ist die Menge der Knoten die einen Knoten kennen und outdegree die Menge der Knoten die ein Knoten kennt. Es wird angenommen, das die indegrees und outdegrees bei bösartigen Knoten höher sind als bei korrekten Knoten. Da sich die bösartigen Knoten untereinander kennen und die bösen Knoten in

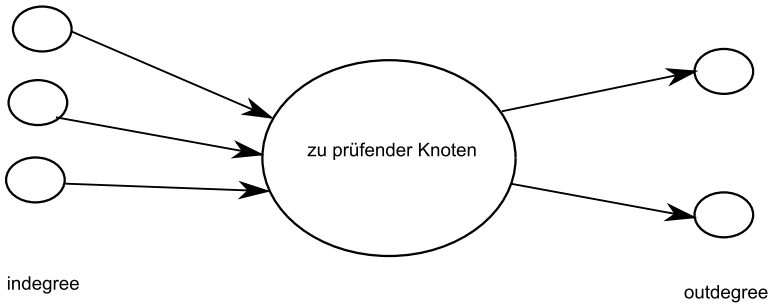

Abbildung 2: Abb.2 Beispiel für indegree und outdegree eines Knoten

vielen Routingtabellen eingetragen sind. Daher wird ein Grenzwert festgelegt. Durch Vergleiche mit den Grenzwerten können bösartige Knoten gezielt erkannt und anschließend gelöscht werden.

Problematisch ist die Identifizierung eines bösartigen Knotens. Jeder Knoten hält eine Liste aller Knoten die ihn kennen, diese Liste wird *backpointer-Liste* genannt. Diese Liste holt sich der anfragende Knoten von einem verdächtigen Knoten. Nun wird diese backpointer List mit der Indegreegrenze verglichen. Hierdurch kann eine höhere Verstrickung eines bösartigen Knoten entdeckt werden. Als erstes prüft der anfragende Nachbarknoten, ob er selber in der backpointer-Liste steht um herauszufinden ob er selber über den Knoten erreichbar ist. Sollte er nicht darin stehen wird dieser Knoten aus der Routingtabelle des anfragenden Knoten gelöscht. Wenn er darin steht, vergleicht er die Anzahl der Einträge der backpointer-Liste mit der Indegreegrenze. Sollte die Anzahl der Einträge höher sein als die Grenze wird der Knoten gelöscht da er bösartig ist. Um Fälschungen beim indegree von „korrekten“ Knoten zu vermeiden können diese durch Kontrolle des outdegrees geprüft werden. Hierfür wird der outdegree mit der Nachbarmenge des „korrekten“ Knotens verglichen. Als erstes prüft der anfragende Knoten wieder ob er selber in der Nachbarschaftsmenge steht, sollte er nicht darin stehen löscht er den Knoten. Wenn er darin vorkommt prüft er nun ob die Nachbarschaftsmenge größer ist als der outdegree, sollte dies der Fall sein führt dies ebenfalls zum Löschen.

Damit diese Prüfungen korrekt funktionieren können, darf der befragte Knoten den anfragenden Knoten nicht kennen. Sollte er den anfragenden kennen, könnte er seine backpointer-Liste oder Nachbarschaftsmenge auf den Anfragenden anpassen, das heißt er löscht so viele Einträge bis er unter den in/outdegree Grenzen des Anfragenden ist und behält die ID des Anfragend [2]. Um anfragende Knoten anonym zu halten nutzt man *Audit-Technik* [2]. Hierfür werden alle Anfragen über Vermittlerknoten geführt. Um Anfragen authentisch zu machen werden diese signiert und mit einem Identifizierer versehen.
Zusammenfassend ist zu sagen, dass diese Verteidigung sowohl in unstrukturierten als auch in strukturierten Netzen funktioniert. Allerdings benötigt sie eindeutige Knotenidentifizierer und einen anonymen Kanal für Anfragen. Ein Nachteil ist, dass Performanceoptimierungen bei dieser Verteidigung nicht funktionieren [2].

4 Andere Angriffe

Es wird nun auf weitere Angriffe und mögliche Verteidigungen eingegangen.

4.1 Routing Angriff

Der Routing Angriff stellt eine Bedrohung für Knoten dar die intial in ein Netzwerk eingeführt werden. Hierbei fragen neue Knoten meistens nach der Routingtabelle eines nahen Knotens. Genau an dieser Stelle sind neue Knoten besonders verwundbar. Sollte der Knoten, der die neue Routingtabelle liefert, bösartig sein, kann dieser die Routingtabelle des neuen Knotens komplett sabotieren, indem er falsche, bösartige oder nicht existierende Einträge übergibt. Zudem kann ein böser Knoten einen neuen Knoten in ein paralleles Netzwerk einführen [3] mit den selben Protokollen usw. allerdings nur mit bösen Knoten besetzt. Eine falsche Einführung kann man nur durch Verwendung einer gesicherten Quelle verhindern. Außerdem kann der eigentliche Routing Mechanismus sabotiert werden vg.Kapitel3.

4.2 Überladen von Knoten

Das Überladen von Knoten kann auch als Denial of Service Angriff gegen einen bestimmten Knoten angesehen werden. Hierbei wird ein Knoten gezielt zum Überladen (funktionsunfähig) gebracht. Dafür schickt ein bösartiger Knoten so lange Müll-Pakete an den Knoten, bis dieser unter der Last zusammenbricht oder das System den Knoten für fehlerhaft erachtet und ihn aus dem System entfernt. Das System macht dies beispielsweise durch eine Ping Anfrage. Ein Ping ist eine Nachfrage nach einem Lebenszeichen welche mit einem Pong beantwortet wird. Sollte ein Knoten gar nicht oder zu spät auf einen Ping antworten löscht das System den Knoten.

4.3 Rapid Joins and Leaves

Beim *Rapid Joins and Leaves Angriff* handelt sich es um einen Angriff der die Aktualisierungs- und Instandhaltungsalgorithmen eines Netzes nutzt bzw. missbraucht. Die meisten Netze reagieren auf einen Beitritt bzw. Ausstieg damit, dass sie die Routingtabellen der Nachbarn aktualisieren. Sollten nun aber mehrere Angreifer an verschiedenen Stellen beitreten und verlassen wird das System dazu gezwungen an allen Stellen die Routingtabellen zu aktualisieren. Das gefährliche an dieser Aktualisierung ist, dass sie besonders belastend für das Netz ist. Die möglicherweise gleichzeitige Aktualisierung vieler Knoten kann zu Ausfällen von Knoten und ganzen Teilsystemen führen. Problematisch ist dabei eine Datenverschiebung aufgrund der Aktualisierungsalgorithmen. Einige Netze verteilen einen Datensatz an mehrere Rechner um Ausfälle zu vermeiden. Jedesmal wenn nun ein Peer beitritt bekommt er einen Datensatz für den er verantwortlich ist und jedesmal wenn ein Peer das Netz verlässt muss ein anderer Peer den Datensatz übernehmen. Dieses Verschieben belastet das Netz erheblich [4].

4.4 Inkonsistenzes Verhalten

Problematisch ist das teilweise Fehlverhalten eines Knotens. Eine Möglichkeit um festzustellen ob ein Knoten bösartig ist, ist es über Knoten Berichte anzufertigen, mit dem Inhalt, ob ein Knoten sich gut oder böse verhält und dann eine Mehrheitsabstimmung aufgrund dieser Berichte zu machen. Allerdings führen diese Berichte zu einem neuen Problem. Sie müssen fälschungssicher sein, sonst erzeugt ein bösartige Knoten viele gute Berichte über sich selbst. Durch digitale Signaturen kann das verhindert werden. Dabei wird der Bericht eindeutig von einem Knoten erzeugt und somit fälschungssicher.

4.5 Fremdnachrichten

Das Verhindern von falschen Antworten auf Anfragen kann ein Problem darstellen. Angenommen es gibt drei Knoten A, B und C. A stellt nun eine Anfrage an C und leitet dabei über B. Angenommen die Anfrage ist eine Rechnung z.B. „3+3=?“. Normalerweise gibt nun B die Anfrage an C weiter, C berechnet das Ergebnis, gibt es B und B gibt Antwort an A weiter. Für den Fall, dass B ein böser Knoten ist, leitet dieser nicht an C weiter. Dieser Knoten kann die Anfrage nehmen und als Ergebnis 4 antworten, dies schickt er an A. Um dieses Problem zu verhindern gibt es wieder die Lösung durch digitale Signaturen. Diese sichern, dass die Anfrage von C beantwortet wird und anschließend nicht gefälscht werden kann. Durch die digitalen Signaturen kann dann A die Signatur verifizieren.

5 Fireflies

Fireflies stellt ein skalierbares Protokoll für ein eindringlings-tolerantes Overlay dar. Es wird angenommen [5], dass ein Eclipse Angriff wirkungslos ist, da Sub-Netze von korrekten Mitgliedern gebildet werden. Das Firflies-Protokoll besteht aus einem Gossip-Protokoll, einem Membership-Protokoll, einer Daten-Struktur und den daraus folgendenen Protokoll-Schritten.

5.1 Gossip-Protokoll

Dieses Protokoll sichert einen broadcast-Kanal. Der broadcast-Kanal muss gesichert werden, damit alle Nachrichten an alle Mitglieder geschickt werden und dass Nachrichten auch nicht gefälscht, sabotiert oder in irgendeiner Art und Weise behindert werden können.

5.2 Membership-Protokoll

Die Grundidee dieses Protokolls ist das gegenseitige Beschuldigen und Überwachen von korrekten Mitgliedern. Das Membership-Protokoll nimmt an, dass jedes Mitglied einen einzigartigen Identifikator hat, der entweder im Zustand korrekt, gestoppt oder byzantinisch ist. Es wird angenommen, dass ein korrektes Mitglied das Protokoll korrekt ausführt. Ein gestopptes führt kein Protokoll aus. Es wird angenommen, dass ein byzantinisches Mitglied auf Beschuldigungen nicht antwortet und im broadcast-Kanal nichts anderes macht als andere Mitglieder zu beschuldigen. Durch das ständige Beschuldigen von byzantinischen Mitgliedern, muss man die Ringe so wählen, dass es nicht zuviel Overhead gibt und der Kanal dadurch zusammenbricht. Außerdem wird vorausgestetzt, dass jedes korrekte Mitglied eine Sicht aller Mitglieder hat, die für eine „lange“ Zeit korrekt waren.

5.3 Datenstruktur

Mitglieder sind in mehreren Ringen angeordnet. Dabei wird angenommen, dass Mitglieder in mehreren Ringen sein können. Die Anzahl der Ringe lässt sich aus der Formel 2t+1 Ringe ableiten. Für t wird eine Wahrscheinlichkeit für die zu erwartenden byzantinischen Mitglieder eingesetzt. Problematisch ist es t, so zu wählen, dass man nicht zu viele und nicht zu wenige Ringe hat. Zu viele Ringe können nur schwer gegen byzantinische Mitglieder vorgehen, zu wenige Ringe haben zur Folge, dass der Overhead durch den broadcast-Kanal zu groß wird [5]. Außerdem wird angenommen, dass jedes Mitglied einen Rang bekommt. Jedes Mitglied überwacht seinen Nachfolger und sendet eine Beschuldigung über das Gossip-Protokoll, sollte er ihm verdächtig vorkommen. Für Beschuldigungen gilt, dass ein Mitglied nur durch ein höher geranktes (Vorgängerknoten) Mitglied beschuldigt werden darf.

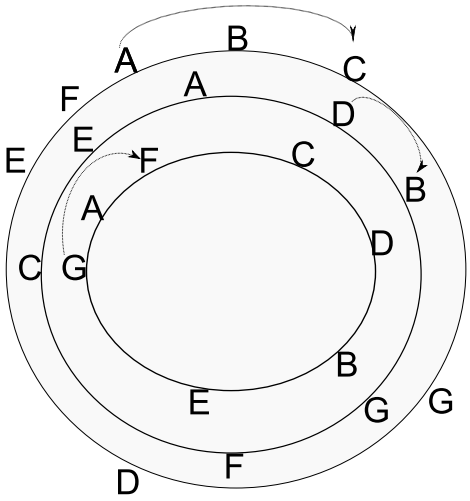

Abbildung 3: Abb.3 Die einzelnen Protokollschritte im Detail.

5.4 Protokoll-Schritte

Wie mit gestoppten Knoten umgegangen wird, beschreiben die folgenden Protokollschritte im Detail.

1. Ein Mitglied erhält eine Notiz für ein anderes Mitglied.
   Angenommen ein Mitglied m1 empfängt eine Notiz, also ein „Lebenszeichen“, für das Mitglied m2. Als erstes vergleicht m1 die Notiz mit seiner eigenen. Sollte die Notiz älter sein als seine eigene, ignoriert er die neue Notiz. Sollte aber die neue Notiz neuer oder genau gleich alt sein, löscht er die Beschwerde die gegen m2 vorliegt und stoppt den Timer.

2. Ein Mitglied verdächtigt ein anderes Mitglied.
   Angenommen m1 verdächtigt m2. Als erstes beobachtet m1 m2, das heißt, er prüft ob m2 das Protokoll korrekt ausführt oder gestoppt ist. Sollte er nun denken m2 ist gestoppt, stellt er eine Beschwerde über den broadcast-Kanal. Alle Mitglieder starten daraufhin einen Timer. Sollte m2 innerhalb der Timer-Zeit keine Notiz an alle Mitglieder schicken, wird m2 als gestoppt betrachtet und gelöscht.

3. Ein Mitglied erhält eine Beschwerde von einem Mitglied.
   Angenommen m1 erhält eine Beschwerde für m2. Als erstes prüft m1 die Beschwerde auf ihrer Korrektheit. Eine Beschwerde muss zum Beispiel von einem höher gerankten Mitglied als m2 erfolgen. Sollte die Beschwerde fehlerhaft sein wird sie ignoriert. Sollte nun m1 bemerken, dass er selbst beschuldigt wird, schickt er eine Notiz an alle Mitglieder über das Gossip-Protokoll, dass die ihren Timer stoppen.

Im Folgenden soll die Arbeitsweisen des Fireflies-Protokoll erklärt werden vg.Abb.3.
Angenommen auf dem untersten Ring möchte G F beschuldigen. Dies geht jedoch nicht, da A zwischen G und F liegt und somit A höhergerankt ist als G. Da die Beschuldigung fehlerhaft ist wird sie ignoriert.
Als nächstes wird auf dem mitleren Ring angenommen das D B beschuldigt. Diese Anschuldigung ist korrekt da niemand zwischen D und B auf dem Ring liegt. Damit starten alle ihren Timer. Falls B kein Lebenszeichen abschickt, wird B nach Ablaufen des Timers von allen Mitgliedern in allen Ringen

gelöscht.
Zuletzt wird auf dem äußeren Ring angenommen, dass A C beschuldigen will. Diese Beschuldigung ist auch korrekt, denn A ist höher gerankt, da B bereits gelöscht wurde. Damit starten alle Mitglieder ihren Timer gegen C. C sendet nun eine Notiz an alle, d.h. alle Mitglieder stoppen ihren Timer gegen C.

5.5 Schluß

Um relativ sicheres routen zu garantieren sind ein einige Punkte zu beachten [3]. Es werden sichere Knotenidentikatoren benötigt um zum Beispiel einen Sybil-Angriff zu verhindern. Desweiteren werden gesicherte Routingtabellen benötigt um beispielsweise einen Eclipse-Angriff zu vermeiden. Als letztes wird sicheres weiterleiten von Nachrichten benötigt um Angriffe wie Fremdnachrichten zu verhindern. Doch jede Verteidigung beinhaltet auch eine Performanceverminderung, die man für Sicherheit zahlen muss.

6 Quellen

Literatur

[1] John R. Douceur: *The Sybil Attack*. Peer-to-Peer Systems: First InternationalWorkshop, 2002.

[2] Atul Singh, Miguel Castro, Peter Druschel, Antony Rowstron: *Defending against Eclipse attacks on overlay networks*.Proceedings of the 11th workshop on ACM SIGOPS European workshop: beyond the PC, 2004.

[3] Miguel Castro, Peter Druschel, Ayalavadi Ganesh, Antony Rowstron, Dan S. Wallach: *Secure routing for structured peer to peer overlay networks*.Fifth Symposium on Operating Systems Design and Implementation, 2002.

[4] Emil Sit, Robert Morris: *Security Considerations for Peer-to-Peer Distributed Hash Tables*.Peer-to-Peer Systems: First InternationalWorkshop, 2002.

[5] Havard Johansen, Andre Allavena, Robbert van Renesse. *Fireflies: Scalable Support for Intrusion-Tolerant Overlay Networks.* Proceedings of Eurosys 2006, Willy Zwaenepoel, ACM European Chapter, April, 2006.

[6] Christian Schindelhauer: *Peer to Peer Netzwerke*. Vorlesung Uni Freiburg Sommersemester 2006.

[7] Christoph Gow,Adrian Leemann, Amir Saddat: *Security in Peer to Peer Systemen*. Powerpoint Vortrag,2006.

[8] Flora Rhea Schreiber.*Sybil*.Warner Books, Auflage: Reissue Juli 1995.