query flood DoS attacks in Gnutella

von Andreas Legrum

basierend auf einem Dokument von
Neil Daswani und Hector Garcia-Molina

Für das Seminar
"Peer-to-peer Information Systems"

Übersicht

1. Einführung

Das Thema des Quellenpapiers waren "Query Flood DoS Attacks in Gnutella", also "anfragenüberflutungsbasierte DoS-Attacken in Gnutella", und wie man auf sie unter Verwendung verschiedener Netzwerkstrukturen und Regeln zu Auswahl von Queries darauf reagieren kann um den Schaden zu minimieren. Um Vergleichswerte für verschiedene Kombinationen von Auswahlkriterien und Topologien zu bekommen wurde ein vereinfachtes Model des Datenflusses der Anfragen erstellt, mit welchem danach mehrere Testläufe stattfanden.
In Abschnitt (2) findet sich Erklärungen zu diesem vereinfachten Model sowie eine kurze Übersicht über die verschiedenen Regeln zur Query-Auswahl.
Im darauf folgenden Kapitel werden die Formalismen zum Messen des durch die Attacken angerichteten Schadens vorgestellt.
Abschnitt (4) befasst sich dann mit den durch die Testläufe erzielten Resultaten, die im Quellenpapier vorgestellt wurden.
Das letzte Kapitel enthält eine kurze Zusammenfassung der Testergebnisse und einige Anmerkungen.

2. Vereinfachtes Model eines Gnutella-P2P-Netzwerkes

Wie zuvor angesprochen befasst sich dieser Abschnitt mit dem vereinfachten Model, wie es von den Autoren entwickelt wurde. Zuerst wird dabei die Struktur und der Nachrichtenfluss vorgestellt, gefolgt von einer Beschreibung der Auswahlregeln und einigen Implementierungsdetails in Bezug auf die Knoten.

2.1 Die vereinfachte Netzwerkstruktur

Gnutella stellt ein Netzwerk aus miteinander verknüpften Superknoten da, die ihrerseits lokal angeschlossene Clients besitzen. Die lokalen Clients produzieren dabei Anfragen nach Dateien, die über die Superknoten zu den Clients anderer Superknoten weitergeleitet werden. Diese Anfragen besitze eine Lebenszeit (TTL) die angibt, wie oft die Anfrage weitergeleitet werden soll.
Im Model sind ein Superknoten und seine lokale Klienten zu einer Einheit zusammengefasst, da sich die Experimente nur mit dem Datenfluss der Anfragen zu Volllastzeiten beschäftigen, so dass man davon ausgehen kann, dass ein Superknoten von seinen lokalen Clients ausreichend viele Anfragen erhält.

2.2 Grundsätze zur Auswahl von Queries

Superknoten haben nur eine beschränkte Rechenkapazität, d.h. sie können nur eine bestimmte Anzahl von Anfragen verarbeiten. Das vorgestellte Model enthält deshalb drei verschiedene Methoden, anhand derer ein Superknoten seine Kapazität verteilt: Reservation Ratio, Incoming Allocation Strategy (IAS) und Drop Strategy(DS).

Die "Reservation Ratio" ist ein Wert zwischen 0 und 1 der angibt, wie viel Rechenleistung den Anfragen lokaler Clients zugeteilt wird, wobei 1 für die totale Zuteilung steht. Sollte für die lokalen Clients weniger Rechenleistung benötigt werden, kommt der Rest anderen Anfragen zu Gute.

Die "Zuweisungsstrategie für Eingehendes" IAS legt fest, wie viele Anfragen eines bestimmten anderen direkt verbundenen Superknotens verarbeitet werden, ausgehen von der vorhandenen Restkapazität.

Im Quellpapier werden zwei verschiedene IASs vorgestellt. Bei der gewichteten IAS werden die Anfragen prozentual gleich abgearbeitet, d.h. das alle Superknoten den gleichen Anteil an Anfragen bearbeitet bekommen. Je mehr Anfragen von einem verbundenen Superknoten aus eingehen, desto mehr werden davon verarbeitet. Im Fall der fraktalen IAS wird die Rechenleistung gleichmäßig auf alle Verbindungen zu anderen Knoten verteilt. Sollte ein Nachbarknoten weniger Leistung verbrauchen, als ihm zusteht, so wird diese Restkapazität nach selbem Muster weiterverteilt.

Nachdem nun feststeht, wie viele Anfragen eines Nachbarknotens bearbeitet werden, wird nun anhand der "Drop Strategy" entschieden, welche gewählt werden. Im Model besteht eine Anfrage aus dem Tupel von Ursprungsknoten und Restlaufzeit (TTL). Diese werden nun in Abhängigkeit von Quelle und TTL in Gruppen eingeteilt. Bei der proportionalen DS werden nun ähnlich der gewichteten IAS alle Gruppen zu prozentual gleichen Teilen bedient. Je mehr Anfragen ihren Ursprung also bei einem bestimmten Knoten haben, desto mehr davon werden bearbeitet. Dem gegenüber steht die 'equal DS', bei der versucht wird, die gleiche Anzahl von Anfragen aus allen Gruppen zu bearbeiten, also entsprechend der fraktalen IAS. Neben diesen beiden Drop-Strategien gibt es u.a. noch die nach TTL ordnenden DS, die Anfragen mit hoher oder niedriger TTL bevorzugen.

2.3 "gute" und "böse" Knoten

Bei der Simulation wird zwischen den "guten" und dem "bösen" Knoten unterschieden. Bei allen Knoten wird die gleiche Rechenleistung angenommen. Die guten Knoten versuchen dabei, die Arbeitsleistung des Netzwerks zu maximieren, indem sie ihre "Reservation Ratio" auf einen optimalen Wert setzen. Dem gegenüber steht der bösen Knoten, die ihre Rechenleistung dazu Nutzen, sinnlose Anfragen zu stellen. Da den Knoten eine unbeschränkte Anzahl von lokalen Anfragen zur Verfügung steht lässt sich der böse Knoten dadurch modellieren, dass er eine 'preservation Ratio' von 1 hat, also nur lokale Anfragen weiterreicht, nicht jedoch solche anderer Superknoten.

3. Messen des Schadens

Der verursachte Schaden lässt sich als Verringerung des angebotenen Services messen. Dabei richtet der Knoten zwei verschiedene Arten von Schaden an: einerseits Schaden dadurch, dass er selbst keine Anfragen weiterleitet (struktureller Schaden), andererseits verbrauchen die von ihm ausgesendeten Informationen unnötig Rechenleistung auf anderen Knoten ('Flut'-Schaden). Beide Sorten von Schaden werden jedoch zusammen gemessen.

Um den Schaden zu berechnen, den ein böser Knoten in einem Netzwerk anrichtet, wurden verschiedene Netzwerktopologien in der Größe von 14-16 Knoten aufgebaut und für 100 Takte simuliert. Der angebotene Service des Netzwerks berechnet sich dabei aus der Summe der zu jedem Takt bearbeiteten Anfragen (abzüglich derjenigen, die ihren Ursprung beim bösen Knoten hatten). Der angerichtete Schaden ergibt sich dabei aus dem Vergleich der Serviceleistung des Netzwerks für den Fall, dass es nur aus guten Knoten besteht und für den Fall, dass einer der Knoten eine Query-flood-Attacke durchführt. Der böse Knoten saß dabei an verschiedenen markanten Punkten des Netzwerks und die angewendeten Strategien wurden ebenfalls gewechselt um Werte für verschiedene Kombinationen zu erhalten.

4. Effektivität der Auswahlkriterien

Die Messungen ergaben folgende Werte für die verschiedenen Zusammensetzungen aus Topologien und Strategien:

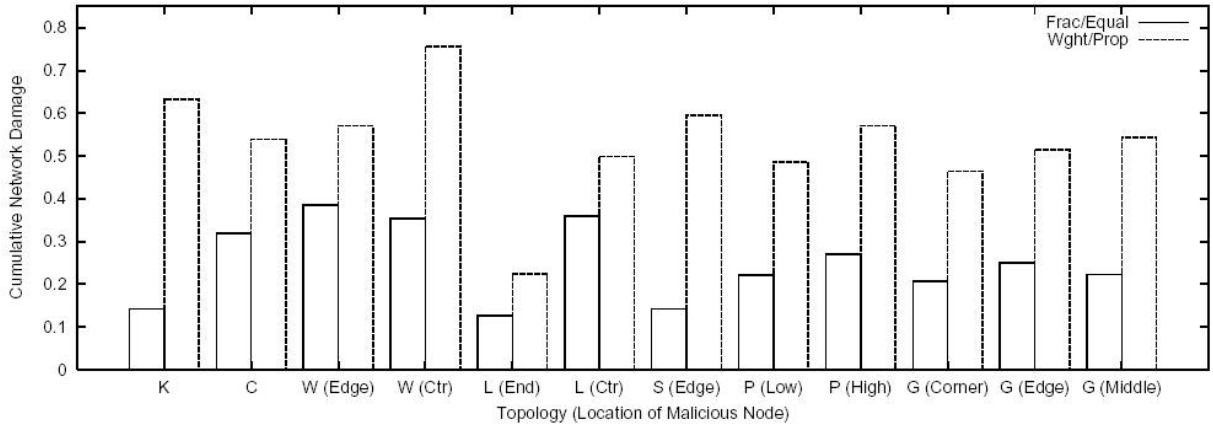
	Fractional				*Weighted*			
Topology (Location)	*Prop*	*Equal*	*PfHighTTL*	*PfLowTTL*	*Prop*	*Equal*	*PfHighTTL*	*PfLowTTL*
Complete	0.143	0.143	0.143	0.143	0.633	0.633	0.633	0.633
Cycle	0.407	0.319	0.319	0.533	0.539	0.459	0.399	0.699
Grid (Center)	0.340	0.243	0.331	0.360	0.545	0.511	0.555	0.685
Grid (Corner)	0.282	0.232	0.266	0.405	0.455	0.372	0.378	0.613
Grid (Edge)	0.310	0.220	0.306	0.429	0.519	0.406	0.433	0.633
Line (Center)	0.393	0.360	0.387	0.457	0.500	0.426	0.458	0.616
Line (End)	0.162	0.126	0.135	0.299	0.225	0.185	0.165	0.366
Power-Law (High)	0.288	0.279	0.307	0.333	0.573	0.550	0.530	0.684
Power-Law (Low)	0.260	0.189	0.227	0.237	0.478	0.423	0.445	0.589
Star (Center)	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
Star (Edge)	0.143	0.143	0.143	0.143	0.595	0.595	0.595	0.595
Wheel (Center)	0.354	0.354	0.354	0.354	0.756	0.776	0.756	0.777
Wheel (Edge)	0.440	0.359	0.387	0.412	0.560	0.508	0.561	0.672

Dabei ist anzumerken, dass struktureller Schaden nur durch eine geeignetere Netzwerktopologie abgefangen werden kann. So tritt bei der kompletten Vernetzung der Knoten keinerlei struktureller Schaden auf, da jeder Knoten direkt mit allen anderen verbunden ist. Jedoch nimmt bei dieser Topologie bei größeren Systemen die Netzwerklast stark zu, so dass sie in solchen Fällen nicht praktikabel ist. Bei der Sterntopologie bricht hingegen das gesamte Netz auf Grund des strukturellen Schadens zusammen, wenn der böse Knoten im Zentrum sitzt. Die weiteren Betrachtungen beziehen sich daher auf die Wirksamkeit der einzelnen Strategien gegen die Anfrageflut selbst, obwohl auch der strukturelle Schaden in die Berechnungen eingegangen ist.

Wie aus der Tabelle leicht ersichtlich ist, tritt bei fraktaler IAS ein teilweise deutlich geringerer Schaden auf als bei der gewichteten IAS, unabhängig von DS und Topologie. Dies liegt daran, dass die nutzlosen Anfragen bei gewichteter IAS eine höhere Chance haben, weitergeleitet zu werden, vor allem dann, wenn der böse Knoten eine große Anzahl an Anfragen aussendet. Im Vergleich zwischen 'equal' und 'proportional' DS ergibt sich das gleiche Bild: auf Grund der Tatsache, dass die 'equal' DS unabhängig von der Anzahl der eingegangenen Anfragen die zu verarbeitenden auswählt verhilft sie zu einem geringeren Schaden. Beim Vergleich der auf TTL basierenden DS stellt man fest, dass das bevorzugen geringer Restlaufzeiten teilweise sogar den angerichteten Schaden zu vergrößern schein. Dies lässt sich dadurch erklären, dass sich die Reichweite dir sinnlosen Anfragen erhöht, da sie mit zunehmender Entfernung eine größere Wahrscheinlichkeit zur Weiterleitung erhalten. Das Bevorzugen hoher Restlaufzeiten hingegen kann zwar den Schaden auch eindämmen, jedoch geht dies zu lasten der Reichweite aller Anfragen, so dass dies meist der Qualität der durch die Anfrage erzielten Ergebnisse schadet.

Topology (Location)	*Frac/Equal*	*Wght/Prop*	*Dmg Red Ftr*
Complete	0.143	0.633	4.4
Cycle	0.319	0.539	1.7
Grid (Center)	0.243	0.545	2.2
Line (Center)	0.360	0.500	1.4
Power-Law (High)	0.279	0.573	2.1
Star (Center)	1.000	1.000	1.0
Wheel (Center)	0.354	0.756	2.1

Bei der Gegenüberstellung der IAS/DS-Paarungen gewichtet/proportional und fraktal/'equal' zeigt sich, dass man den Schaden im günstigsten Fall um einen Faktor von über vier reduzieren kann und dass die zweite Paarung immer am Besten abschneidet. Dies ist, obigen Ausführungen folgend, auch nicht verwunderlich, da die Kombination von gewichteter IAS und proportionaler DS besonders anfällig ist, wenn ein einzelner Knoten eine große Anzahl an Anfragen aussendet.



In obiger Grafik sind die Daten nochmals nebeneinander gestellt, die Namen der Topologien wurden abgekürzt (K=Complete, C=Cycle, W=Wheel, L=Line, S=Star, P=Power-Law, G=Grid). Es ist deutlich sichtbar, dass eine Veränderung der Topologie alleine im Schnitt keine große Verbesserung bringt, gepaart mit den richtigen Strategien jedoch durchaus Sinn machen kann. Auch kann man sehen, dass böse Knoten, die näher am Zentrum eines Netzwerkes sitzen, einen größeren Schaden anrichten können.

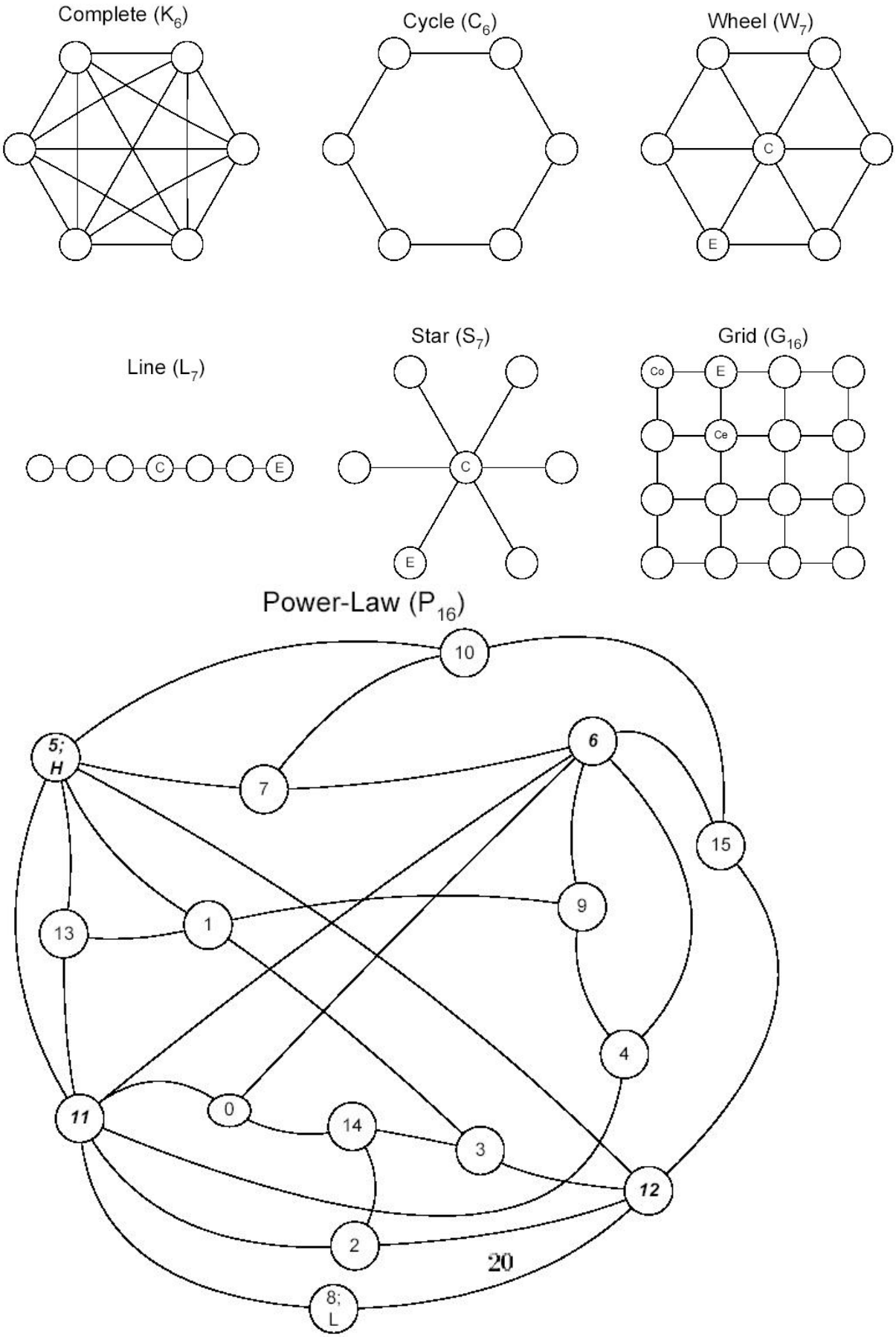
5. Zusammenfassung

Die Daten zeigen, dass Gnutella-Netzwerke bei schlechter Konfiguration sehr anfällig für DoS-Attacken sind. Diese lassen sich zwar nicht verhindern, jedoch kann der durch sie angerichtete Schaden durch eine Anpassung des Systems teilweise deutlich reduziert werden. Verfahren, die gegen die Ausnutzung von Schwächen resistenter sind, sind jedoch nur ein erster Schritt. Es müssen Verfahren gefunden werden, die verhindern, dass böse Knoten sich dem Zentrum des Netzwerkes nähern und die solche Knoten ausfindig machen und vom Netzwerk trennen.
Bei den Daten des Dokumentes ist zu beachten, dass sie auf simulierten Systemen ermittelt wurden, die u.a. die Netzwerkbandbreite außer Acht lassen. Es ist fraglich, ob sich die Ergebnisse 1:1 übertragen lassen, jedoch dürften zumindest die Tendenzen auch im realen Netzbetrieb zutreffen.

6. Quell-Dokument

Das Quell-Dokument ist zu finden unter:
http://dbpubs.stanford.edu:8090/pub/2002-26

Anhang a:


Complete (K_6)
Cycle (C_6)
Wheel (W_7)
C
E
Line (L_7)
C
E
Star (S_7)
C
E
Grid (G_{16})
Co
E
Ce
Power-Law (P_{16})
10
5;
H
6
7
15
9
13
1
4
11
0
14
3
12
2
20
8;
L