

Analyse des OpenPGP Web of Trust

Studienarbeit
von

Alexander Ulrich

Lehrstuhl für Rechnernetze und Internet
Wilhelm-Schickard-Institut für Informatik
Fakultät für Informations- und Kognitionswissenschaften
Universität Tübingen

Erstgutachter: Prof. Dr. Peter Hauck
Betreuer Mitarbeiter: Dipl.-Inform. Ralph Holz

Bearbeitungszeit: 08. Monat 2009 – 01. Monat 2010

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Tübingen, den ???. ?????? 200?

Inhaltsverzeichnis

| | | |
|----------|--|-----------|
| 1 | Grundlagen | 1 |
| 1.1 | Kryptographie mit öffentlichen Schlüsseln | 1 |
| 1.1.1 | Prinzip | 1 |
| 1.1.2 | Authentisierung von Schlüsseln | 1 |
| 1.1.2.1 | Zentrale PKI | 1 |
| 1.1.2.2 | Web of Trust | 1 |
| 1.2 | PGP/GnuPG | 1 |
| 1.2.1 | Geschichte von PGP und GnuPG | 1 |
| 1.2.2 | Eigenschaften/Fähigkeiten der Implementierungen | 1 |
| 1.2.3 | Das GnuPG-Vertrauensmodell | 1 |
| 1.2.4 | Soziale Komponente des Web of Trust | 3 |
| 1.3 | Der OpenPGP-Standard (RFC4880) | 4 |
| 1.3.1 | Paketformat v4 | 4 |
| 1.3.2 | Unterschiede v3 | 4 |
| 1.4 | Graphentheorie allgemein | 4 |
| 1.5 | Netzwerkanalyse | 4 |
| 1.5.1 | Netzwerkstatistiken | 4 |
| 1.5.2 | Communities | 4 |
| 1.5.3 | Modularity | 4 |
| 1.6 | Verwandte Arbeiten | 4 |
| 1.6.1 | Analysen des OpenPGP-Web of Trust | 4 |
| 1.6.2 | Community-Strukturen allgemein | 4 |
| 2 | Methoden und Materialien | 5 |
| 2.1 | Warum eigene Software? | 5 |
| 2.2 | Design und Implementierung | 7 |
| 2.2.1 | Der SKS Keyserver | 7 |
| 2.2.2 | Eigene Software | 7 |
| 2.2.2.1 | Datenextraktion | 8 |
| 2.2.2.2 | Datenauswertung | 10 |
| 2.3 | Community-Analyse | 12 |
| 2.3.1 | Communities in gerichteten Graphen | 14 |
| 3 | Ergebnisse und Diskussion | 17 |
| 3.0.2 | Datensatz | 17 |
| 3.1 | Allgemeine Merkmale des Netzwerkes | 18 |
| 3.1.1 | Starke Zusammenhangskomponenten | 18 |
| 3.1.2 | Netzwerkstatistiken der grössten starken Zusammenhangskomponente | 20 |

| | | |
|----------|--|-----------|
| 3.1.3 | Robustheit | 22 |
| 3.1.4 | Nützlichkeit | 23 |
| 3.2 | Eigenschaften einzelner Schlüssel, zeitliche Entwicklung | 25 |
| 3.2.1 | Zeitliche Entwicklung | 25 |
| 3.2.2 | Public-Key- und Hashalgorithmen | 27 |
| 3.3 | Zusammenhangskomponenten und Communities | 27 |
| 3.3.1 | Communities | 27 |
| 3.3.2 | Inhaltliche Analyse der Zusammenhangskomponenten | 41 |
| 4 | Zusammenfassung und Ausblick | 43 |
| | Literaturverzeichnis | 45 |

1. Grundlagen

1.1 Kryptographie mit öffentlichen Schlüsseln

1.1.1 Prinzip

Klassische Kryptographie basiert auf einem *Geheimnis* oder *privaten Schlüssel*, der beiden Kommunikationspartnern bekannt ist.

1.1.2 Authentisierung von Schlüsseln

Zweck von PKI oder Web of Trust ist es, die Authentizität der Bindung von Schlüssel/Zertifikat und Identität zu etablieren. Und das eben auch dann, wenn man selbst diese Bindung nicht überprüfen kann.

1.1.2.1 Zentrale PKI

1.1.2.2 Web of Trust

1.2 PGP/GnuPG

1.2.1 Geschichte von PGP und GnuPG

1.2.2 Eigenschaften/Fähigkeiten der Implementierungen

1.2.3 Das GnuPG-Vertrauensmodell

Öffentliche PGP-Schlüssel werden oft nicht in einer Weise übergeben, die die sofortige Verifikation des Schlüssels zulässt, beispielsweise bei einem persönlichen Treffen. Stattdessen werden Schlüssel häufig per E-Mail, über einen Keyserver oder andere elektronische Wege ausgetauscht. Überprüfung der Authentizität eines Schlüssels ist deswegen von grosser Bedeutung.

Ein Schlüssel wird von GnuPG genau dann als gültig (*valid*) betrachtet, wenn er die folgenden Bedingungen erfüllt:

1. Der Schlüssel wurde von ausreichend vielen *gültigen* Schlüsseln unterschrieben, d.h. er wurde mindestens entweder von
 - dem Besitzer des Schlüsselrings selbst (d.h. von einem Schlüssel mit *ultimate trust*) unterschrieben
 - mindestens N gültigen Schlüsseln, denen voll vertraut wird, unterschrieben
 - mindestens M gültigen Schlüsseln, denen geringfügig vertraut wird, unterschrieben
2. Eine Signaturkette wird nur verwendet, wenn sie ausgehend vom Besitzer des Schlüsselrings maximal die Länge L hat.

Ein Schlüssel, der von weniger voll bzw. geringfügig vertrauenswürdigen Schlüsseln als notwendig unterschrieben wurde, wird als eingeschränkt gültig (*marginally valid*) angesehen. Allerdings werden Schlüssel dieser Kategorie von GnuPG genauso wie nicht gültige Schlüssel behandelt.

GnuPG verwendet in der Standardeinstellung die Werte $N = 1$, $M = 3$ und $L = 5$. Damit wird *ein* Zertifikat, dass von einem Schlüssel mit vollem Vertrauen ausgestellt wurde, als ausreichend betrachtet. Für Schlüssel, denen nur geringfügig vertraut wird, ist ein einzelnes Zertifikat nicht ausreichend. Dieses muss noch durch 2 weitere solche Zertifikate bestätigt werden.

GnuPG erlaubt es einem Anwender, die Parameter N , M und L selbst zu setzen und damit seine Sicherheitsanforderungen umzusetzen. Je höher beispielsweise die notwendige Anzahl von Signaturen, um so kleiner ist der Schaden, den eine einzelne fehlerhafte Signatur anrichten kann. Ist die maximale Pfadlänge auf einen kleinen Wert begrenzt, so ist auch die maximale Anzahl der Signaturen auf dem Pfad kleiner, die potentiell fehlerhaft sein können. Andererseits verringert sich damit die Anzahl der Signaturen (Pfade im Web of Trust), die für die Verifizierung benutzt werden können, und damit die Anzahl verifizierbarer Schlüssel. Es muss also eine Abwägung zwischen dem Sicherheitsbedürfnis des Nutzers und der praktischen Benutzbarkeit getroffen werden.

Abbildung 1.1 gibt ein Beispiel für die Berechnung der Schlüsselgültigkeit unter Verwendung der Standard-Parameter $N = 1$, $M = 3$ und $L = 5$. Die Schlüssel B , H und K wurden direkt von A , dem Besitzer des Schlüsselrings, unterschrieben und sind deshalb voll gültig. Da L , M , und N von K unterschrieben wurden und dieser über volles Vertrauen verfügt, sind sie ebenfalls voll gültig. O sowie J sind voll gültig, da sie jeweils von drei Schlüsseln mit geringfügigem Vertrauen unterschrieben wurden. I und P wurden dagegen jeweils nur von zwei Schlüsseln mit geringfügigem Vertrauen unterschrieben und sind deshalb nicht voll sondern nur eingeschränkt gültig. G wurde zwar von einem voll gültigen Schlüssel mit vollem Vertrauen unterschrieben. Allerdings überschreitet die Signaturkette zu A die maximale Länge von 5 und wird deshalb nicht betrachtet.

Ein öffentlicher Schlüssel, der anhand dieser Regeln nicht als authentisch verifiziert werden kann, kann trotzdem zur Verschlüsselung und zur Verifizierung von Signaturen verwendet werden. Allerdings warnt GnuPG in diesem Fall vor der Verwendung.

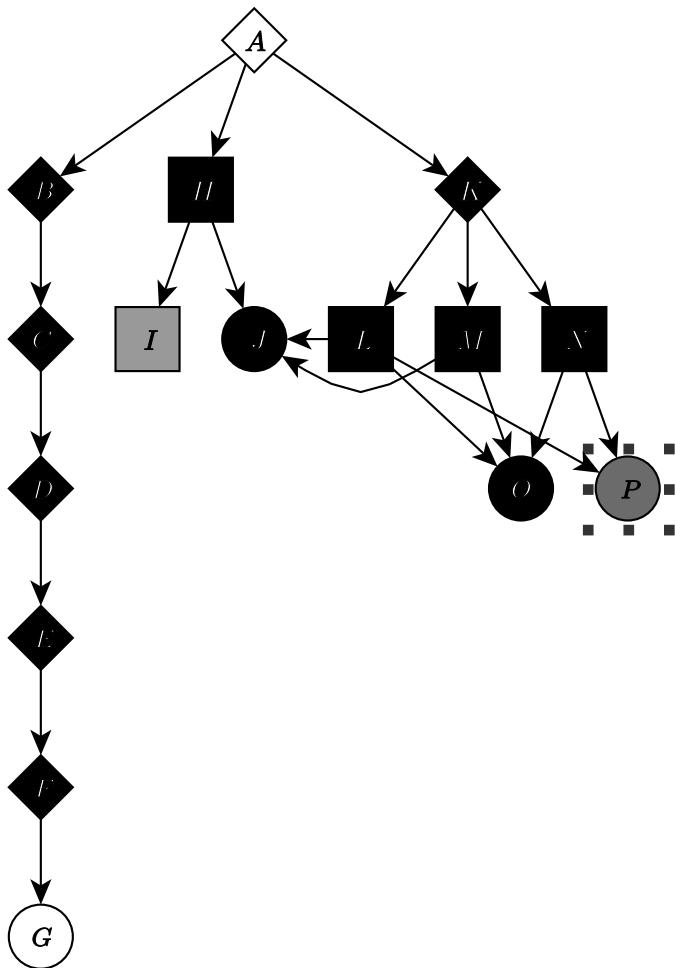


Abbildung 1.1: Beispiel für die Berechnung der Gültigkeit von Schlüsseln

1.2.4 Soziale Komponente des Web of Trust

PGP-CAs (CACert?, CT) erwähnen, Crypto-Kampagne der CT

In einer Reihe von Open-Source-Projekten spielen PGP und das Web of Trust eine wichtige Rolle. Im Debian-Projekt beispielweise werden PGP-Schlüssel unter anderem benutzt, um alle bereitgestellten Softwarepakete durch den zuständigen Entwickler signieren zu lassen. Dazu muss jeder Debian-Entwickler über einen Schlüssel verfügen, der von mindestens einem anderen Debian-Entwickler verifiziert und unterschrieben wurde. Auf diese Weise bildet sich bereits innerhalb des Projekts ein Web of Trust. PGP scheint eine wichtige Rolle in der Kultur des Projekts zu spielen.

1.3 Der OpenPGP-Standard (RFC4880)

1.3.1 Paketformat v4

1.3.2 Unterschiede v3

1.4 Graphentheorie allgemein

1.5 Netzwerkanalyse

1.5.1 Netzwerkstatistiken

1.5.2 Communities

1.5.3 Modularity

$$Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{d_i d_j}{2m} \right) \delta(c_i, c_j) \quad (1.1)$$

Seit der Arbeit von Newman wurde eine Vielzahl weiterer Methoden zur Erkennung von Communities vorgestellt. An dieser Stelle werden 3 Methoden näher beschrieben, die in dieser Arbeit verwendet werden. Einen Gesamtüberblick bietet ein Übersichtsartikel von Fortunato [Fort10].

1.6 Verwandte Arbeiten

1.6.1 Analysen des OpenPGP-Web of Trust

1.6.2 Community-Strukturen allgemein

2. Methoden und Materialien

2.1 Warum eigene Software?

Das bereits in Abschnitt 1.6 erwähnte *wotsap*-Projekt berechnet täglich die Struktur des Web of Trust. Die Daten werden für eine Web-Applikation benutzt, die grundlegende Statistiken über Schlüssel berechnet und Pfade zwischen Schlüsseln graphisch darstellt. Zusätzlich werden die Daten aber auch für weitere Analysen zur Verfügung gestellt. In diesem Abschnitt wird begründet, warum für die vorliegende Arbeit nicht auf diese Daten zurückgegriffen wurde, sondern die Datenextraktion selbst vorgenommen wurde.

Ein Ziel dieser Arbeit ist es, die Struktur des Web of Trust abseits der grössten starken Zusammenhangskomponente zu untersuchen. *wotsap* berechnet allerdings nur die Struktur eben dieser Zusammenhangskomponente. Schlüssel, die nicht in dieser Komponente enthalten sind, werden nicht beachtet. *wotsap* beginnt bei mehreren sehr gut vernetzten Schlüsseln, die sicher in der MSCC liegen. Von diesen ausgehend werden die Signaturen in der Art einer Breitensuche (rückwärts) verfolgt. Aufgrund dieser Methode scheint es mit vertretbarem Aufwand nicht möglich, die Extraktion auf alle Schlüssel auszudehnen.

Der Anwendungszweck der *wotsap*-Daten ist ausschliesslich die strukturelle Analyse des Netzes. Die über die reine Struktur hinausgehenden Daten, die über Schlüssel und Signaturen gespeichert werden, sind dabei auf ein Minimum reduziert: Für Schlüssel werden ausschliesslich die KeyID und die primäre UserID gespeichert, für Signaturen der Cert level und die Schlüssel. Diese Reduktion erlaubt zwar eine sehr kompakte Speicherung der Daten, macht es aber für eine Auswertung weiterer Eigenschaften von Schlüsseln und Signaturen unbrauchbar. Das verwendete Dateiformat ist ausserdem recht unflexibel und lässt eine Speicherung weiterer Daten nur mit grösserem Aufwand zu.

Die *wotsap*-Daten beinhalten nur die zum jeweiligen Zeitpunkt gültigen Schlüssel und Signaturen. Dieser „Schnappschuss“ reicht für die strukturelle Analyse des Graphen aus. Zeitliche Entwicklungen, beispielsweise die Grösse des Datenbestandes, die Verwendung bestimmter Verschlüsselungs- und Signaturalgorithmen und die Entwicklung einzelner Komponenten können damit aber nicht nachvollzogen werden.

Die *wotsap*-Methode liefert also nicht die im Rahmen dieser Arbeit benötigten Daten. Eine Anpassung der Software würde auf eine komplette Neuimplementierung hinauslaufen. Außerdem beruht die Extraktion der Daten bei *wotsap* auf der veralteten und kaum mehr benutzten *PKS*-Keyserver-Implementierung (siehe Abschnitt 2.2.1).

Darüber hinaus ist allerdings der *wotsap*-Datensatz fehlerhaft. Diese Fehler werden sowohl durch Fehler in der Implementierung als auch durch einen fehlerhaften Datenbestand auf dem verwendeten Keyserver verursacht: Die grösste starke Zusammenhangskomponente die durch XXX berechnet wurde (MSCC-1), enthält mit dem Stand vom 02.12.2009 ca. 44900 Schlüssel, während der *wotsap*-Datensatz (MSCC-2) vom gleichen Tag nur ca. 42130 Schlüssel enthält. Die Differenz zwischen den Datensätzen ergibt einerseits, dass ca. 1000 Schlüssel in MSCC-1 fehlen, die in MSCC-2 vorhanden sind. Eine stichprobenartige Analyse von 20 Schlüsseln zeigt, dass diese Schlüssel überwiegend durch Signaturketten an die MSCC angebunden sind, die aufgrund von widerrufenen Schlüsseln oder Schlüsselteilen unterbrochen sind. Dabei treten u.a. Signaturen auf komplett widerrufenen Schlüsseln und Signaturen auf widerrufenen UserIDs auf. *wotsap* benutzt GnuPG, um die OpenPGP-Pakete eines Schlüssels zu parsen. Der Code zum Parsen der GnuPG-Ausgabe ist fehlerhaft und führt dazu, dass *wotsap* Widerruffsignaturen fälschlicherweise nicht beachtet.

Auf der anderen Seite sind ca. 3980 Schlüssel zwar in MSCC-1 vorhanden, nicht aber in MSCC-2. Um den Grund dafür zu finden, wurde für 10 zufällig ausgewählte Schlüssel aus dieser Menge eine Signaturkette (d.h. ein Pfad) zu einem Schlüssel gesucht, der sowohl in MSCC-1 als auch in MSCC-2 vorhanden ist. Ebenfalls wurde eine Kette in der anderen Richtung gesucht. Die Signaturen dieser Ketten wurden mittels GnuPG kryptographisch verifiziert, um sicherzustellen, dass sie gültig sind. Kann für beide Richtungen jeweils eine Kette erfolgreich verifiziert werden, so ist der betreffende Schlüssel per definitionem in der MSCC. Im vorliegenden Fall konnte das für alle Schlüssel der Stichprobe gezeigt werden. Der Fehler liegt also wiederum bei *wotsap*, dass diese Schlüssel fälschlicherweise ausschliesst. Als Ursache dafür ergab sich in allen betrachteten Fällen der fehlerhafte Bestand des Keyserver *wwwkeys.ch.pgp.net*, der von *wotsap* verwendet wird. Die Schlüssel wurden von *wotsap* nicht in die MSCC-2 übernommen, weil Teile der dazu notwendigen Signaturketten (komplette Schlüssel oder einzelne Signaturen) auf dem Keyserver nicht vorhanden sind. Besonders häufig traten dabei Schlüssel auf, deren Ablaufdatum durch eine neue Selbstsignatur verlängert wurde. Dieses Signaturpaket fehlte auf *wwwkeys.ch.pgp.net*, wodurch der betreffende Schlüssel fälschlicherweise als abgelaufen betrachtet wurde. Diese fehlenden Teile sind aber auf allen Keyservern des *SKS*-Verbundes vorhanden. Da die Ursache in allen untersuchten Fällen die gleiche war, kann angenommen werden, dass der Fehler systematisch ist und diesen Teil der Diskrepanz zwischen den Datensätzen vollständig erklärt. Als Ursache kommen eine mangelhafte Synchronisation zwischen *wwwkeys.ch.pgp.net* und dem *SKS*-Netzwerk sowie Fehler in der auf *wwwkeys.ch.pgp.net* verwendeten *PKS*-Version in Frage.

Die Natur der Fehler legt nahe, dass hauptsächlich solche Schlüssel fehlen bzw. fälschlich einbezogen wurden, die nur über eine sehr geringe Anzahl von redundanten Pfaden zur bzw. von der MSCC verfügen. Gäbe es mehr redundante Pfade, so hätten einzelne fehlerhafte Informationen (fehlende bzw. fälschlicherweise als gültig betrachtete Schlüssel oder Signaturen) einen geringeren Einfluss.

Die Anzahl fehlender bzw. fälschlich einbezogener Schlüssel (ca. 9% fehlend, ca. 2% fälschlich einbezogen relativ zu MSCC-1) ist so gross, dass qualitative Fehler in der Graphenstruktur zu erwarten sind. Insbesondere Aussagen über die Verteilung von Knotengraden und ähnliche Aussagen anhand dieser Daten sind mit Vorsicht zu geniessen. Eine Reihe von Arbeiten verwendet die *wotsap*-Daten als Beispiel für ein empirisches soziales bzw. Small-World-Netzwerk [BrSc06] [HeGu09] [Dell07]. Sofern die Ergebnisse dieser Arbeit auf experimentellen Resultaten aufbauen, die mit diesen Daten gewonnen wurden, sollten sie daher mit korrekten Daten überprüft werden. Der Keyserver *wwwkeys.ch.pgp.net* sollte von Anwendern nicht mehr benutzt werden.

2.2 Design und Implementierung

Dieser Abschnitt beschreibt die im Rahmen dieser Arbeit erstellte Software zur Extraktion und Analyse des Web of Trust. Abschnitt 2.2.1 beschreibt zunächst die Keyserver-Software *SKS*, auf die der Extraktionsteil aufbaut. Abschnitt 2.2.1 beschreibt dann die entwickelte Software selbst und begründet das Design.

2.2.1 Der SKS Keyserver

SKS (Synchronizing Key Server) löst die bis dahin verbreitetste Keyserver-Software *PKS* (Public Key Server) ab und wird inzwischen auf so gut wie allen Keyservern benutzt. Mit *pgp.mit.edu* wurde im Juli 2009 der letzte grosse Keyserver auf *SKS* umgestellt. *PKS* benutzt ein auf E-Mails basierendes Protokoll zur Synchronisation zwischen Keyservern, das ausgesprochen ineffizient ist. Außerdem kommt es mit einigen Bestandteilen des aktuellen OpenPGP-Standards wie z.B. *PhotoIDs* und mehreren Unterschlüsseln nicht zurecht. *PKS*-Keyserver können über ein Webinterface, ein E-Mail-Interface und das auf HTTP basierende HKP-Protokoll¹ abgefragt werden.

Im Unterschied dazu kommunizieren *SKS*-Keyserver zur Synchronisation direkt (ohne Umwege über Mailserver) miteinander. Dazu wird ein binäres „Gossip“-Protokoll benutzt, dass auf einem effizienten Algorithmus zum Abgleich von Datensätzen beruht [MiTZ03]. *SKS* unterstützt alle Bestandteile des OpenPGP-Standards.

SKS ist in der Sprache Objective Caml (OCaml) implementiert und benutzt die Datenbank Berkeley DB als Datenablage. Ein Datenbankeintrag besteht dabei aus einem kompletten OpenPGP-Key, d.h. einer Reihe von OpenPGP-Paketen. *SKS* ist in zwei Prozesse aufgeteilt: der *db*-Prozess beantwortet Datenbank-Abfragen für die verschiedenen Abfragemöglichkeiten (Webinterface, HKP) während der *recon*-Prozess für den Abgleich der Datenbank mit anderen Keyservern zuständig ist.

2.2.2 Eigene Software

Die im Rahmen dieser Arbeit entwickelte Software besteht aus zwei Teilen: Der Extraktionsteil liest Schlüssel aus der Datenbank eines Keyservers aus, parst sie, reduziert sie dabei auf die benötigten Daten und speichert sie in einer Datei ab. Die reduzierten Daten werden in einer SQL-Datenbank (PostgreSQL FIXME cite) abgelegt und können dort abgefragt werden. Der zweite Teil besteht aus einer Reihe von Werkzeugen zur Analyse dieser Daten entsprechend der Zielsetzung dieser Arbeit.

¹<http://tools.ietf.org/html/draft-shaw-openpgp-hkp-00>

2.2.2.1 Datenextraktion

Die Extraktion der Daten ist recht zeitaufwendig. Auf der zur Verfügung stehenden Hardware (FIXME Poolrechner Hardware) werden ca. XX Stunden benötigt. Durch die Aufteilung kann die Extraktion der Daten einmalig auf dem Keyserver vorgenommen werden, während die Daten anschliessend auf beliebigen Rechnern zur Verfügung stehen.

Der Extraktionsteil wurde direkt in die SKS-Software integriert. Dadurch ergeben sich mehrere Vorteile:

- Das Interface von SKS für den Datenbankzugriff kann wiederverwendet werden
- SKS enthält einen rudimentären OpenPGP-Parser, d.h. eine Reihe von Funktionen, die die Byte-Struktur einzelner OpenPGP-Pakete sowie die Paket-Struktur des Schlüssels parsen und die darin enthaltenen Informationen in Datenstrukturen zugänglich machen. Durch die Verwendung dieser Funktionen kann auf eine eigene Implementierung eines OpenPGP-Parsers verzichtet werden.
- Die in SKS definierten Datenstrukturen für die Auswertung von OpenPGP-Paketen und weitere Hilfsfunktionen, beispielsweise für den Umgang mit OpenPGP-Fingerprints, können verwendet und erweitert werden

Derzeit läuft der Extraktionsteil in Form eines eigenen Prozesses, der einmalig den kompletten Datenbestand ausliest. Dieser Teil könnte aber auch direkt in den *db*-Prozess integriert werden, so dass der laufende Keyserver konstant neue oder geänderte Schlüssel reduziert und in der SQL-Datenbank ablegt. Auf diese Weise könnte ein ständig aktueller Datenbestand ohne den Aufwand des vollständigen Auslesens realisiert werden. Diese Möglichkeit wurde im Rahmen dieser Arbeit nicht verfolgt, kann aber mit geringem Aufwand realisiert werden.

Da der Extraktionsprozess nur lesend auf die SKS-Datenbank zugreift, kann er die Daten auslesen, während die *recon*- und *db*-Prozesse laufen. Eine Unterbrechung des SKS-Betriebes ist nicht notwendig.

Als Sprache für die Implementierung wurde OCaml ausgewählt. Im Extraktionsteil bestand dafür aufgrund der Integration in SKS keine Wahl. Der Analyseteil wurde ebenfalls in OCaml implementiert. Da die dort verwendeten Daten ausschliesslich aus der SQL-Datenbank stammen und dort nur primitive Datentypen (Strings, Ganz- und Fliesskommazahlen) gespeichert werden, können aber für diesen Teil auch problemlos andere Sprachen verwendet werden.

Der Extraktionsteil iteriert über alle in der SKS-Datenbank enthaltenen Schlüssel. Jeder Schlüssel wird durch die in SKS enthaltenen Funktionen geparsst. Der Extraktionsteil FIXME muss dann noch

- Entscheiden, ob der Schlüssel zurückgezogen oder abgelaufen ist. Dazu werden die Selbstsignaturen auf den einzelnen UserIDs betrachtet. Falls zu einer UserID mehrere Selbstsignaturen vorliegen, wird entsprechend der Empfehlung im OpenPGP-Standard nur die aktuellste verwendet. Liegt ein Rückrufzertifikat vor oder ist der Schlüssel abgelaufen, wird das Ablauf- bzw. Rückzugsdatum gespeichert.

- Fremdsignaturen sammeln. Dazu wird jede UserID betrachtet und die darauf angebrachten Signaturen ohne Duplikate gespeichert. Hier wird ebenfalls nur die neueste Signatur verwendet. Handelt es sich bei der neuesten Signatur um eine Rückrufsignatur, wird das Datum des Rückrufs gespeichert und zusätzlich die nächstältere Signatur gesucht, auf die sich der Rückruf bezieht.
- Den Schlüssel und jede Fremdsignatur auf die benötigten Daten reduzieren. Für den Schlüssel sind dies (falls vorhanden) Rückzugs- und Ablaufdatum, die KeyID, die Liste aller UserIDs und die Information, welche davon die primäre UserID ist, das Erstellungsdatum des Schlüssels und der verwendete Public-Key-Algorithmus sowie dessen Schlüssellänge.

Eine Fremdsignatur wird reduziert auf die KeyID des Unterzeichners FIXME, den Zertifizierungslevel, das Erstellungsdatum der Signatur, falls vorhanden Ablauf- und Rückzugsdatum, der verwendete Signaturalgorithmus und der verwendete Public-Key-Algorithmus.

Grundsätzlich werden Schlüssel nur dann verworfen, wenn sie nicht parsebar, d.h. keine Abfolge gültiger OpenPGP-Pakete sind. Schlüssel, die zurückgezogen oder abgelaufen sind, werden unter Angabe des jeweiligen Datums trotzdem gespeichert. Auf diese Weise ist sichergestellt, dass der Datensatz möglichst vollständig ist. Für einen beliebigen Zeitpunkt stehen alle dann gültigen Schlüssel und Signaturen zur Verfügung. Es kann also gewissermaßen ein „Snapshot“ des Web of Trust zu einem beliebigen Zeitpunkt analysiert werden.

Sind alle Schlüssel extrahiert, werden noch solche Signaturen entfernt, deren erstellender Schlüssel im Datenbestand nicht vorhanden ist. Außerdem werden für Signaturen, die von einem Unterschlüssel erstellt wurden, die KeyID des signierenden Schlüssels auf die des „Oberschlüssels“ geändert. Dies ist notwendig, weil das hier verwendete Datenmodell keine Informationen über Unterschlüssel enthält.

Erwähnt werden muss, dass der Parse-Vorgang nur prüft, ob die Paketfolge eines Schlüssels dem OpenPGP-Standard entspricht. Es wird keinerlei kryptographische Verifizierung der Selbst- und Fremdsignaturen eines Schlüssels vorgenommen. Grundsätzlich können ohne Probleme Signaturpakete auf einem Schlüssel angebracht und diese auf einem Keyserver veröffentlicht werden, auch wenn der Ersteller nicht über das private Schlüsselmaterial für die kryptographische Signatur verfügt. Keyserver verifizieren keine Signaturen, so dass der Signaturteil des Pakets mit beliebigem Inhalt gefüllt werden kann. GnuPG und PGP verifizieren natürlich Signaturen, so dass eine solches Signaturpaket dort nicht verwendet wird und damit kein wirkliches Angriffspotential bietet. Eine Möglichkeit für den Extraktionsteil bestünde darin, die Signaturen jedes OpenPGP-Schlüssels mit GnuPG verifizieren zu lassen. Dagegen sprechen zwei Gründe: Einerseits wäre der Aufruf von GnuPG sehr zeitaufwendig. Für jeden Schlüssel müssten der Schlüssel in einen GnuPG-Schlüsselring eingefügt werden. Zusätzlich müssten noch alle Schlüssel, die den jeweiligen Schlüssel signiert haben, einzeln in der SKS-Datenbank gesucht und in den Schlüsselring eingefügt werden, um die Signaturen verifizieren zu können. Letztendlich kostet der Aufruf von GnuPG selbst Zeit. Für die Anzahl der hier verarbeiteten Schlüssel (2,6 Millionen) scheint dieser Ansatz daher ungeeignet. Sicherlich sind Schlüssel mit defekten Signaturpaketen auf den Keyservern vorhanden. Allerdings müsste deren Anzahl sehr gross sein, um signifikante Änderungen in der Struktur des Graphen und den

statistischen Auswertungen der Schlüsseleigenschaften zu erreichen. Eine grosse Zahl solcher Pakete scheint aber unwahrscheinlich, da zumindest unter Sicherheitsaspekten ein Angreifer damit keinen offensichtlichen Gewinn erreichen kann.

Die so extrahierten Daten werden auf einem beliebigen Recher in einer SQL-Datenbank (PostgresQL) abgelegt. Auf diese Weise kann darauf verzichtet werden, eigene Selektionsmechanismen zu implementieren. Stattdessen kann die gewünschte Datenmenge einfach als SQL-Abfrage formuliert werden. Auf diese Weise ergibt sich eine deutlich flexiblere Abfragemöglichkeit. Die Ablage in der Datenbank ist zwar etwas langsamer als die Daten im Speicher zu halten. Andererseits müssen die Daten aber nicht jedesmal komplett in den Speicher geladen und dort gehalten werden. Ausserdem werden effiziente Indexstrukturen der Datenbank genutzt und müssen nicht selbst implementiert werden.

Die Tabellen sind wie folgt strukturiert: TODO

Die Zuordnung einzelner Schlüssel zu ihren starken Zusammenhangskomponenten erfolgt über die Tabelle *component_ids*. Diese wird erst in einem separaten Schritt befüllt, nachdem die Komponentenstruktur berechnet wurde.

```
(SELECT signer, signee
  FROM sigs INNER JOIN keys ON sigs.signer = keys.keyid
 WHERE
    (keys.revoketime IS NULL OR keys.revoketime > $timestamp)
    AND (keys.exptime IS NULL OR keys.exptime > $timestamp)
    AND (sigs.revoketime IS NULL OR sigs.revoketime > $timestamp)
    AND (sigs.exptime IS NULL OR sigs.exptime > $timestamp))
INTERSECT
(SELECT signer, signee
  FROM sigs INNER JOIN keys ON sigs.signee = keys.keyid
 WHERE
    (keys.revoketime IS NULL OR keys.revoketime > $timestamp)
    AND (keys.exptime IS NULL OR keys.exptime > $timestamp)
    AND (sigs.revoketime IS NULL OR sigs.revoketime > $timestamp)
    AND (sigs.exptime IS NULL OR sigs.exptime > $timestamp))"
```

Abbildung 2.1: Abfrage aller zum Zeitpunkt \$timestamp gültigen Signaturen

Als Beispiel gibt Abb. 2.1 ein SQL-Statement an, mit dem alle gültigen (d.h. nicht zurückgezogenen oder abgelaufenen) Signaturen zu einem bestimmten Zeitpunkt abgerufen werden können.

Wie beschrieben wird in einem Durchlauf der gesamte Datenbestand extrahiert. Aufgrund der Integration in SKS wäre es auch problemlos möglich, den Extraktionsanteil nicht als eigenständigen Prozess zu implementieren, sondern den *db*-Prozess entsprechend zu erweitern, so dass jeder neue oder geänderte Schlüssel direkt in der SQL-Datenbank abgelegt wird. Auf diese Weise könnte ein jederzeit aktueller Datenbestand erhalten werden, ohne dass er ständig komplett neu berechnet werden muss.

2.2.2.2 Datenauswertung

Der Teil zur Datenauswertung besteht aus einer Reihe von unabhängigen Werkzeugen, die die jeweiligen Analyseaufgaben implementieren. Diese sind als Kommandozeilenprogramme in eigenen Prozessen realisiert. Die Aufgaben der einzelnen Werkzeuge sind in Tab. 2.1 aufgeführt.

| | |
|----------------------------|---|
| basic-properties-mpi | Verteilung von Eingangs- und Ausgangsgraden, Komponentengrößen, Nachbarschaften, Durchmesser, Radius, durchschnittliche Pfadlängen (parallelisiert mittels MPI) |
| betweenness-mpi | Berechnung der Betweenness-Zentralität (parallelisiert mittels MPI) |
| clustering-coefficient-mpi | Berechnung des Clustering Coefficient (parallelisiert mittels MPI) |
| export | Export des Graphen in verschiedene Dateiformate (igraph, Cytoscape) |
| meta-graph | Zeichnung der Struktur der starken Zusammenhangskomponenten |
| db-scc-information | Befüllt die SQL-Datenbank mit der Zuordnung von Knoten zu Zusammenhangskomponenten |
| dump-sql | Legt die extrahierten Daten einmalig in der SQL-Datenbank ab |
| investigate-component | Untersucht einzelne Zusammenhangskomponente in Bezug auf Herkunft der UserIDs (Domains) und den Zeitraum der Entstehung der Signaturen |
| mscc-size | Entwicklung der Schlüsselanzahl der grössten starken Zusammenhangskomponente |
| simple-stats | Statistiken über die Verwendung bestimmter Schlüsseleigenschaften (Algorithmen, Schlüssellängen usw.) |
| time | Entwicklung der Verwendung von Signatur- und Public-Key-Algorithmen und deren Schlüssellängen |

Tabelle 2.1: Foobar

2.3 Community-Analyse

In Abschnitt 1.2.4 wurden Mechanismen beschrieben, die zur Entstehung des Web of Trust beitragen. Ausgehend davon kann die Annahme getroffen werden, dass die Vernetzung wesentlich von zwei Faktoren beeinflusst wird: Teilnehmer vernetzen sich einerseits aufgrund ihrer Zugehörigkeit zu einer *sozialen Gruppe*. Dabei kann es sich um Open-Source-Projekte wie z.B. Debian, akademische Einrichtungen und Firmen handeln, aber auch um eine Gruppe von Freunden oder Bekannten. Andererseits vernetzen sich Teilnehmer auf Keysigning-Parties mit einer relativ grossen Anzahl Benutzer, die ihnen nicht unbedingt bekannt sind und mit denen sie keine starke Gruppenzugehörigkeit verbindet. Offensichtlich ist allerdings, dass diese Mechanismen sich nicht gegenseitig ausschliessen. Die bereits in 1.2.4 beschriebene Bandbreite von Keysigning-Parties reicht von Ad-hoc-Veranstaltungen mit wenigen Teilnehmern, die durchaus der selben Institution angehörig sein können, bis zu grossen, formalisierten Veranstaltungen auf Konferenzen und Messen mit etlichen Teilnehmern.

Es soll nun untersucht werden, inwiefern sich diese postulierten Entstehungsmechanismen in der Struktur, also den topologischen Eigenschaften, des Web of Trust wiederfinden. Angenommen wird, dass sich ein hoher Anteil der Teilnehmer primär anhand dieser beiden Mechanismen vernetzen und der Anteil an Signaturen zwischen Teilnehmern, die nicht durch eine Keysigning-Party oder eine soziale Gruppe entstanden ist, deutlich geringer ist. In diesem Fall ist zu erwarten, dass sich soziale Gruppen und die Ergebnisse von Keysigning-Parties als Gruppen von Knoten im Netzwerk wiederfinden, die untereinander eine hohe Anzahl von Kanten hat, d.h. *dicht* vernetzt ist, während sich zu Knoten ausserhalb der Gruppe nur relativ wenige Kanten finden. Diese Eigenschaft entspricht exakt der in Abschnitt 1.5.2 beschriebenen Definition von *Communities*. Es erscheint als Methode deshalb sinnvoll, mit vorhandenen Methoden zur Community-Erkennung eine Zerlegung des Graphen zu berechnen und dann zu untersuchen, inwiefern sich die einzelnen so berechneten Communities entsprechend der Annahme auf soziale Gruppen bzw. Keysigning-Parties abbilden lassen. Beispielsweise könnten sich eine (oder mehrere) Communities ergeben, die von Mitgliedern des Debian-Projekts dominiert werden.

Dazu sind Kriterien nötig, um eine berechnete Community einer sozialen Gruppe, einer Keysigning-Party oder beidem zuzuordnen. Für Keysigning-Parties ist anzunehmen, dass die Signaturvorgänge zwischen den Teilnehmern einer KSP innerhalb eines eng begrenzten Zeitraums nach Stattfinden der KSP vorgenommen werden. Als einfaches Kriterium für die Erkennung einer Keysigning-Party kann also eine starke zeitliche Korrelation der Signaturvorgänge zwischen der Mehrzahl der Community-Mitglieder verwendet werden. Konkret wird hier eine Community als Produkt einer KSP betrachtet, wenn 80% der Signaturen zwischen den Mitgliedern innerhalb eines Monats vorgenommen wurden.

Die Zuordnung zu sozialen Gruppen wird dadurch erschwert, dass – abgesehen von den Schlüsseln und Signaturen selbst – keine empirischen Daten über die Gruppenzugehörigkeit vorliegen. Die einzigen Daten, die eine (primitive) Zuordnung erlauben, sind die in den UserIDs enthaltenen E-Mail-Adressen. Über die Top-Level-Domain (TLD) kann ein User einem Land zugeordnet werden, sofern es sich nicht um eine der generischen TLDs (com, org, net) handelt. Über die Second-Level-Domain kann

versucht werden, einen Nutzer einer Institution zuzuordnen. Ein User, der beispielsweise über eine E-Mail-Adresse mit der Domain debian.org verfügt, ist sicherlich ein Mitglied des Debian-Projekts. Hilfreich dabei ist, dass Teilnehmer dazu tendieren, alle ihre Adressen auf ihren Schlüsseln zu vermerken, so dass viele Schlüssel mehrere UserIDs und E-Mail-Adressen haben (siehe Abschnitt 3.2). Dadurch werden die verfügbaren Informationen über Gruppenzugehörigkeiten erhöht. Adressen verbreiteter “Freemail”-Anbieter wie Gmail, GMX und Yahoo sowie Adressen von Internet Service Providern dürfen dabei natürlich nicht verwendet werden, weil ihre Verwendung nichts über eine Gruppenzugehörigkeit aussagt.

Ähnlich wie für Keysigning-Parties werden wieder einfache Schwellwerte verwendet, um eine Community einer sozialen Gruppe, also einer Domain, zuzuordnen: Eine Community kann einer Domain *zugeordnet* werden, wenn mindestens 80% der Mitglieder eine UserID mit dieser Domain haben. Ausserdem wird eine Community von einer Domain *dominiert*, wenn ein erheblicher Anteil – konkret mindestens 40% – der Mitglieder eine UserID mit dieser Domain haben.

Die Schwellwerte wurden nicht anhand empirischer Daten festgelegt sondern drücken nur ein intuitives Verständnis für die überwiegende Mehrzahl der Mitglieder einer Gruppe bzw. einen erheblichen Anteil der Mitglieder einer Gruppe aus.

Wie bereits in Abschnitt 1.5.2 beschrieben, existiert – insbesondere für ungerichtete Graphen – eine Vielzahl von Methoden zur Zerlegung eines Graphen in Communities. Dabei hat sich bislang kein “bester” Algorithmus herauskristallisiert, auch wenn einige Algorithmen für eine Reihe von Benchmark-Graphen bessere Ergebnisse liefern als andere [LaFo09]. Das übliche Gütemass zum Vergleich von Community-Zerlegungen eines Graphen, dessen tatsächliche Community-Struktur nicht bekannt ist, ist der Modularity-Wert (siehe Abschnitt 2.3). Um zu vermeiden, dass eine vergleichsweise schlechte Einteilung durch einen Algorithmus die Ergebnisse verfälscht, wurden mehrere Algorithmen zur Community-Erkennung auf ungerichteten Graphen verwendet, um ihre Ergebnisse vergleichen zu können: Fast-Modularity (FM) von Newman et al., sowie der Algorithmus von Blondel et al. (LP) (siehe Abschnitt 1.5.2). Die Auswahl wurde aus folgenden Gründen getroffen:

- Fast-Modularity war einer der ersten praktikablen Algorithmen zur Community-Erkennung und wird weiterhin häufig verwendet. Allerdings liefert er in der Literatur durchgängig schlechtere Ergebnisse als moderne Algorithmen [Fort10] und wird deshalb hier nur als Vergleichsbasis verwendet.
- Der Algorithmus von Blondel et al. erzielte im Vergleich von Lancichinetti et al. [LaFo09] mit die besten Ergebnisse.

Ein weiteres Auswahlkriterium war, dass für diese Algorithmen Implementierungen öffentlich mit Sourcecode verfügbar sind. Ausserdem existieren Algorithmen, die zwar für kleinere Graphen bessere Ergebnisse liefern als die hier verwendeten, aufgrund ihrer Berechnungskomplexität für grosse Graphen wie den hier vorliegenden nicht geeignet sind.

Eine Zuordnung von einer Person zu genau einer sozialen Gruppe scheint unrealistisch. Statt dessen ist davon auszugehen, dass eine Person im Allgemeinen zu

mehreren Institutionen oder Gruppen zugehörig ist. Ebenso ist es möglich, dass sie sich einerseits im Kontext einer sozialen Gruppe vernetzt, andererseits aber auch an Keysigning-Parties teilnimmt. Eine solche multiple Zugehörigkeit sollte sich im Netzwerk durch eine Mitgliedschaft eines Knotens in mehreren Communities ausdrücken. Wie in 1.5.2 erwähnt, existiert eine Klasse von Algorithmen, die ein Netzwerk in *überlappende* Communities zerlegt, so dass ein Knoten Teil von mehreren Communities sein kann. Um zu untersuchen, inwiefern dieser Überlappungseffekt die Zuordnung von Knoten zu sozialen Gruppen und KSPs beeinflusst, wurde zusätzlich noch eine überlappende Zerlegung mit dem Algorithmus COPRA [Greg10], einer Verallgemeinerung von Label-Propagation auf berechnet. Dabei handelt es sich um eine Verallgemeinerung des bereits erwähnten Algorithmus Label-Propagation. COPRA erwartet einen Parameter v , der angibt, zu wie vielen Communities ein Knoten gehören kann. Um hier das Optimum bezüglich der Modularity zu finden, wurden Zerlegungen mit Werten $v = 2, \dots, 15$ berechnet. Zum Vergleich wurde eine Zerlegung mit $v = 1$ berechnet, die keine Überlappung erlaubt. Da COPRA nicht-deterministisch ist, wurden für jeden Wert 10 Berechnungen durchgeführt, um die Abweichung zu ermitteln.

Während Modularity die Güte einer Zerlegung nur bezogen auf die Struktur des Graphen misst, drücken die Zuordnungszahlen die Güte einer Partitionierung in Bezug auf die oben formulierte Hypothese über die Entstehung des Netzwerks aus. Wenn diese Hypothese zutrifft, ist eine Zerlegung in Bezug darauf “besser”, wenn eine grössere Anzahl von Communities einer sozialen Gruppe oder Keysigning-Party zugeordnet werden kann. Es kann nicht vorrausgesetzt werden, dass eine Zerlegung mit hoher Modularity automatisch auch eine Zerlegung mit hohen Zuordnungszahlen ist, dass sich also die real existierenden Communities direkt in der Struktur des Graphen wiederfinden.

2.3.1 Communities in gerichteten Graphen

Der Graph des Web of Trust ist inhärent gerichtet, da eine Signatur nur in eine Richtung gilt. Die meisten verfügbaren Methoden zur Community-Erkennung setzen im Gegensatz dazu einen ungerichteten Graphen voraus. Methoden, die auf gerichteten Graphen arbeiten, sind erst in jüngster Zeit vorgestellt worden [LeNe08] [RoBe08]. Der bis jetzt am häufigsten verwendete Ansatz für die Community-Erkennung in gerichteten Netzwerken ist es, die Richtung der Kanten zu ignorieren und den Graphen als ungerichteten Graphen zu behandeln. Auch die Arbeiten, die den Web of Trust-Graphen unter Community-Gesichtspunkten analysieren oder ihn als Benchmark für den Vergleich von Algorithmen verwenden, gehen so vor [BnPSDGA04]. Auch wenn dieses Vorgehen akzeptable Resultate liefern kann, ist doch klar, dass durch das Ignorieren der Richtung ein Informationsverlust stattfindet, der die Ergebnisse negativ beeinflussen kann. In der vorliegenden Arbeit wurde dieser übliche Ansatz des Ignorierens der Kantenrichtung verwendet, da die Methoden für ungerichtete Netzwerke besser etabliert und Implementationen dieser Algorithmen einfacher zugänglich sind. Im Fall des Web of Trust ist allerdings ein erheblicher Anteil der verbundenen Knotenpaare nur in eine Richtung verbunden, d.h. es existiert keine “Gegenkante” (siehe Abschnitt 3.1). Damit ist auch der Informationsverlust durch die Reduzierung erheblich. Es ist daher naheliegend, dass eine Community-Zerlegung des Graphen unter Berücksichtigung der Richtungsinformationen eine Zerlegung berechnen könnte, die die Struktur des Graphen besser wiedergibt und auch unter

dem Gesichtspunkt der Zuteilung zu sozialen Gruppen und Keysigning-Parties bessere Resultate liefert. Um den Einfluss der Richtung zu überprüfen, wurde eine Zerlegung mit dem Algorithmus “infomap” von Rosvall et al. [RoBe08] berechnet.

3. Ergebnisse und Diskussion

In Abschnitt 3.0.2 wird zunächst der berechnete Datensatz allgemein charakterisiert.

3.0.2 Datensatz

Die Extraktion der hier verwendeten Daten wurde auf einem vom Verfasser betriebenen Keyserver am 02.12.2009 vorgenommen. Die Datenbank des Keyservers war zu diesem Zeitpunkt mit dem Rest des Keyserver-Netzwerkes vollständig abgeglichen. Der berechnete Datensatz enthält 2725504 Schlüssel und 1145337 zugehörige Signaturen, wobei hier keine Selbstsignaturen enthalten sind. 52000 Schlüssel wurden während der Extraktion als defekt verworfen. Eine Stichprobe von 30 verworfenen Schlüssel ergab, dass diese auch von GnuPG nicht akzeptiert werden, weil sie entweder über keine UserID-Pakete verfügen oder aus anderen Gründen keine standardkonformen OpenPGP-Schlüssel darstellen. Von den nicht-defekten Schlüsseln sind 417163 Schlüssel abgelaufen und 100071 Schlüssel wurden zurückgezogen. Die in Relation zur Anzahl der Schlüssel niedrige Anzahl von Signaturen weist schon darauf hin, dass ein erheblicher Teil der (gültigen) Schlüssel nicht oder kaum vernetzt ist. In der Tat verbleiben nach Abzug von Schlüsseln, die weder ein- noch ausgehende Signaturen haben, nur 325410 gültige Schlüssel und 816785 zugehörige gültige Knoten, die den Graphen ausmachen. Die grosse Mehrheit der im Keyserver-Netzwerk vorhandenen Schlüssel ist also komplett unvernetzt und nimmt von vorneherein nicht am Web of Trust teil. Die Überprüfung der Authentizität dieser Schlüssel anhand öffentlich verfügbarer Informationen ist nicht möglich. Die Besitzer dieser Schlüssel haben – sofern sie die Schlüssel überhaupt einsetzen – ebenfalls keine Möglichkeit, die Authentizität anderer Schlüssel zu prüfen. Über die Gründe für diese geringe Vernetzung kann hier nur spekuliert werden. Möglich ist etwa, dass die Benutzer schlichtweg keine Notwendigkeit in der Authentifizierung von Schlüsseln sehen, weil ihnen Man-in-the-middle und ähnliche Angriffe nicht bekannt sind, oder das im Web of Trust verwendete Modell zur Verifizierung von Schlüsseln zu komplex erscheint. Es kann selbstverständlich nicht ausgeschlossen werden, dass die Verifizierung über Signaturen läuft, die nicht auf öffentliche Keyserver geladen wurden. Beachtet werden muss auch, dass der Datensatz keine präzise Angabe über die momentane Anzahl von PGP-Benutzern erlaubt, da sich die Schlüssel über einen Zeitraum von 20 Jahren angesammelt haben.

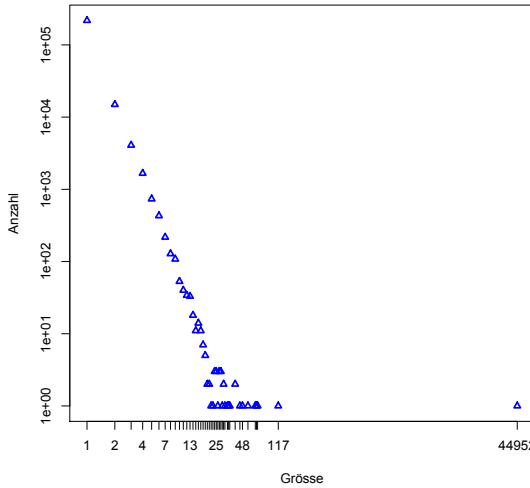


Abbildung 3.1: Größenverteilung der starken Zusammenhangskomponenten

3.1 Allgemeine Merkmale des Netzwerkes

Der so erhaltene Graph ist ein *gerichteter* Graph. Er ist *schlicht*, d.h. er enthält keine *Selbstkanten* und keine *Mehrfachkanten*. Dies ergibt sich, da während der Extraktion Selbstsignuren nicht beachtet und bei mehreren Signaturen des gleichen Schlüssels nur die neueste verwendet wurde.

3.1.1 Starke Zusammenhangskomponenten

Der Graph wurde zunächst in seine 240382 starken Zusammenhangskomponenten zerlegt. Die Verteilung der Komponentengrößen in Abbildung 3.1 zeigt dabei zwei Extreme: Es existiert eine einzelne gigantische Komponente mit ca. 45000 Knoten. Demgegenüber steht eine geringe Anzahl von kleinen Komponenten bis zur Grösse 3 , insbesondere aber über 100000 einelementige Komponenten, also einzelne Knoten, die nur *entweder* ein- oder ausgehende Kanten haben, und über 10000 zweielementige Komponenten, also durch zwei Kanten verbundene Knotenpaare. Ein erheblicher Anteil der überhaupt vernetzten Knoten ist also wiederum nur sehr wenig vernetzt.

Abbildung 3.2 zeigt die Struktur der Zusammenhangskomponenten bis zur Grösse 6, die mit anderen Komponenten verbunden sind. Es ergibt sich eine sternförmige Struktur, bei der die kleineren Komponenten fast ausschliesslich mit der grössten Komponente verbunden sind und es so gut wie keine weitere Vernetzung gibt. Um zwei starke Zusammenhangskomponenten verschmelzen zu lassen, ist nur genau eine Kante in jede Richtung nötig. Es kann also davon ausgegangen werden, dass die Komponenten nur durch einzelne Kanten, die eben nur in eine Richtung verlaufen können, verbunden sind. Daraus ergibt sich, dass die Mitglieder der kleinen Komponenten nur eine sehr geringe Signaturaktivität aufweisen können, da andernfalls die Wahrscheinlichkeit für das Verschmelzen von Komponenten hoch wäre und die Anzahl sehr kleiner Komponenten geringer sein müsste.

Die *Nützlichkeit* des Web of Trust für die Teilnehmer, deren Schlüssel nicht in der grössten starken Zusammenhangskomponente enthalten sind, ist gering: Die Menge der Schlüssel, die von einem Teilnehmer anhand von Signaturketten verifiziert

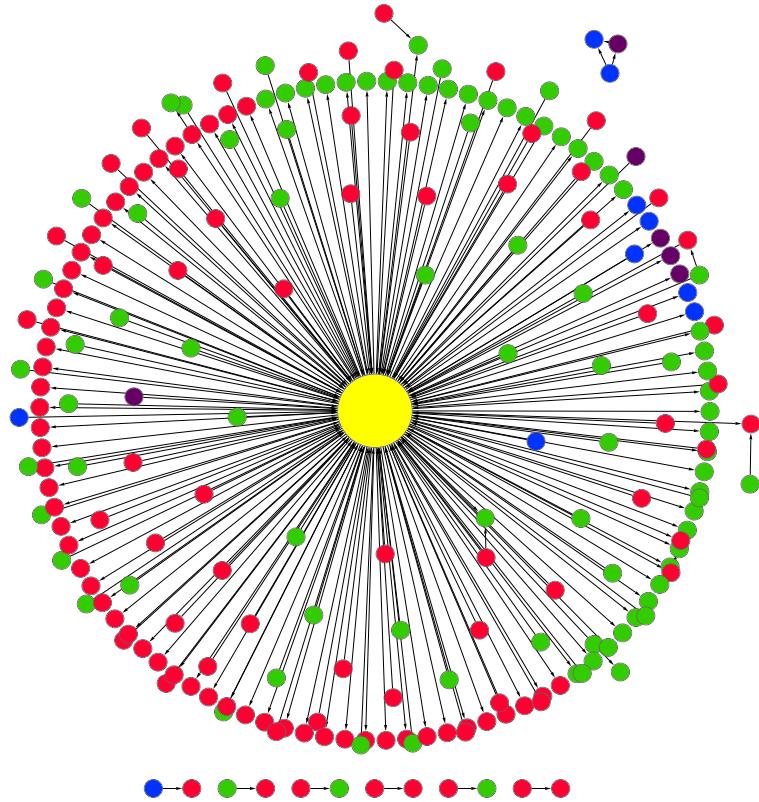


Abbildung 3.2: Struktur der starken Zusammenhangskomponenten bis zur Grösse 6 (rot = Grösse 6-10, grün = Grösse 11-20, blau = Grösse 21-30, violett = Grösse ≥ 31 , gelb = MSCC)

werden kann, beschränkt sich zunächst auf die eigene Komponente und ist damit sehr klein. Zwar gibt es auch – wie oben gezeigt – Vernetzung zwischen kleineren Komponenten und der grössten Komponente. Es existieren ca. 18000 Knoten, die Knoten aus der grössten Komponente direkt über eine einzige Kante erreichen können. Allerdings muss hier beachtet werden, dass das in PGP/GnuPG verwendete Verifizierungsmodell die Länge der verwendbaren Signaturketten begrenzt – in der Standardeinstellung von GnuPG auf die maximale Länge 5 (siehe Abschnitt 1.2.3). Schon innerhalb der grössten Komponente sind viele kürzeste Pfade länger als dieses Maximum (siehe Abschnitt 3.1.2). Durch den zusätzlichen Schritt zu einem Knoten innerhalb der grössten Komponente verlängert sich der Pfad. Insbesondere wenn es sich bei diesen Knoten um schwach vernetzte Knoten handelt, die am “Rand” der grössten Komponente liegen, ist die Menge der auf Pfaden benutzbarer Länge erreichbaren Knoten beschränkt. Kann die grösste Komponente nur indirekt über andere Knoten erreicht werden, reduziert sich diese Menge noch weiter. Gleicher gilt auch für die Erreichbarkeit der Menge von ca. 92000 Knoten, zu denen eine Kante von Knoten der grössten Komponente aus besteht. Beachtet werden muss allerdings die Existenz von zentralen Certificate Authorities im eigentlich dezentralen Web of Trust. Mit den drei seit 1997 im Rahmen der “Krypto-Kampagne” der Zeitschrift c’t (siehe Abschnitt 1.2.4) verwendeten Zertifizierungsschlüsseln wurden insgesamt 23813 derzeit gültige Schlüssel unterschrieben. Von diesen liegen nur 2578 Schlüssel innerhalb der grössten Komponente. Wenn also nur dem verwendeten Zertifizierungsprozess und damit diesen drei Zertifizierungsschlüsseln vertraut wird,

sind immerhin etwa 20000 Schlüssel ausserhalb der grössten Komponente verifizierbar. Es zeigt sich hier auch die Flexibilität des Web of Trust-Konzeptes, dass die Integration von zentralen Komponenten in das eigentlich dezentrale Netz erlaubt.

Insgesamt lassen diese Daten den Schluss zu, dass ausschliesslich die grösste starke Zusammenhangskomponente Schlüssel von Personen enthält, die sich durch regelmässige Signaturaktivitäten am Web of Trust beteiligen und es zur Verifizierung von Schlüsseln benutzen. Eine weitergehende Analyse der in den kleineren Zusammenhangskomponenten enthaltenen Schlüssel wird in Abschnitt 3.3.2 vorgenommen.

3.1.2 Netzwerkstatistiken der grössten starken Zusammenhangskomponente

Die weitere Untersuchung der Topologie des Graphen konzentriert sich auf die grösste starke Zusammenhangskomponente. Starke Zusammenhangskomponenten als Einheit der Betrachtung machen Sinn, weil die Verwendung des Graphen für die Verifizierung von Schlüsseln gerade die Erreichbarkeit voraussetzt und weil eine Reihe von Netzwerkstatistiken nur definiert sind, wenn zwischen allen Paaren von Knoten Pfade existieren. Da im vorherigen Abschnitt argumentiert wurde, dass die grösste starke Zusammenhangskomponente die einzige ist, die in grösserem Massstab vernetzt ist und in der Signaturaktivitäten stattfinden, scheint eine Untersuchung der restlichen Komponenten nicht sinnvoll.

Der induzierte Teilgraph der grössten starken Zusammenhangskomponente besteht aus 44952 Knoten und 442960 gerichteten Kanten. Dieser Teilgraph ist deutlich grösser als die bisher in der Literatur verwendeten PGP-Netzwerke: ein gerichtetes Netzwerk von ca. 12000 Knoten bei [CaBH02] und ein Netzwerk von 10700 Knoten, bei dem alle einseitigen Signaturen gelöscht wurden bei [BnPSDGA04] und [Greg10]. Beide Netzwerke stellen die grösste starke Zusammenhangskomponente dar und wurden im Jahr 2001 extrahiert.

Der *Reciprocity*-Wert eines gerichteten Graphen gibt den Anteil von Kanten an, die in beide Richtungen verlaufen. Für die grösste Komponente ist dieser Wert 0,510. Das bedeutet, dass es für eine gegebene Kante eine Chance von 51% gibt, dass eine entsprechende “Gegenkante” in umgekehrter Richtung existiert. Die Reciprocity eines zufälligen Graphen, der über die *gleiche Gradsequenz* wie die grösste Komponente verfügt, hat im Vergleich eine Reciprocity von nur 0,006. Dass der reale Wert deutlich grösser ist, ist nicht verwunderlich: Das Web of Trust ist eben kein Produkt eines zufälligen Prozesses, sondern von konkreten Mechanismen, nämlich gegenseitigen Signierungen von Schlüsseln. Eher erstaunlich ist, dass der Wert nicht noch höher ist, dass also nicht noch mehr Signaturen auf Gegenseitigkeit beruhen. In Abschnitt 1.2.4 wurde dargestellt, dass die Signierung bei Keysigning-Parties und privaten Treffen üblicherweise gegenseitig verläuft, dass also beide Signaturpartner den Schlüssel des jeweils anderen unterschreiben. Es ist eine Reihe von Gründen denkbar, die dazu führen, dass der Anteil einseitiger Signaturen so hoch ist: Eine Rolle könnten dabei Certificate Authorities spielen. Diese signieren zwar eine Vielzahl von Schlüsseln, empfangen im Allgemeinen aber selbst kaum oder keine Signaturen. Außerdem können Signaturen aus verschiedenen Gründen nicht vorgenommen werden oder nicht auf Keyserver hochgeladen werden. Schlussendlich kann nicht davon ausgegangen werden, dass hinter jeder Signatur ein sinnvoller Prozess

der Identitätsüberprüfung steht. Teilnehmer könnten etwa andere Schlüssel “zum Experimentieren” unterschreiben. PGP und GnuPG zeigen eine Warnung an, wenn die E-Mail-Signatur eines Schlüssels überprüft wird, der nicht als *valide* eingestuft wird. Teilnehmer könnten dann versucht sein, Schlüssel zu unterschreiben, deren Authentizität sie nicht überprüft haben, um diese Warnung zu unterdrücken, selbst wenn dadurch der Sinn des Web of Trust untergraben wird.

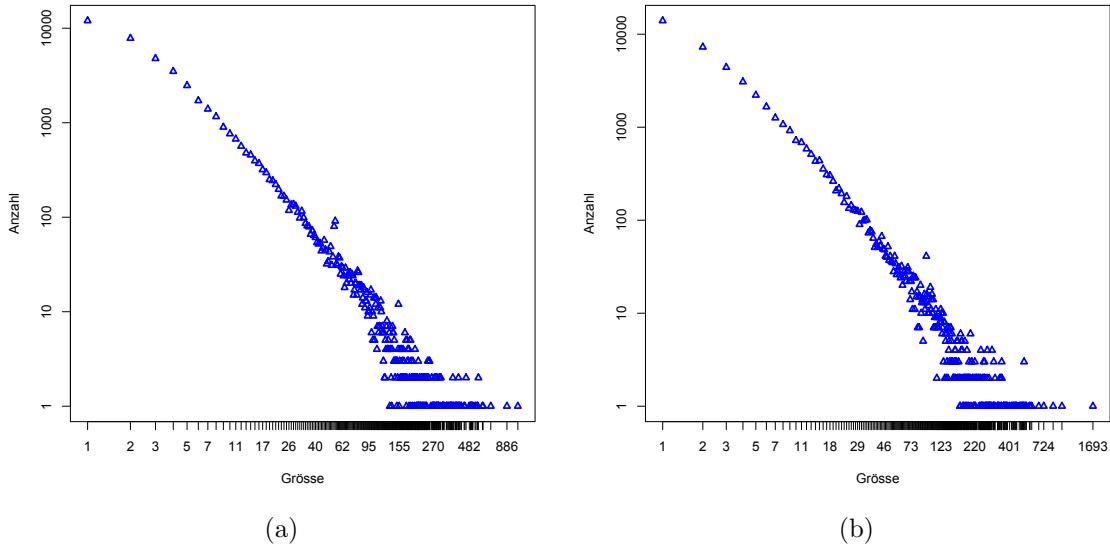


Abbildung 3.3: Verteilung der eingehenden (a) und ausgehenden (b) Knotengrade in der grössten starken Zusammenhangskomponente

(Kein reines powerlaw).

Unabhängig von der konkreten Verteilung kann aber festgehalten werden, dass es eine (kleine) Anzahl von Knoten mit sehr hohem Grad gibt. (weiter ausführen, was das für das Netzwerk bedeutet – Hubs usw.) Orientieren an (Newman – Email, Bele – Email). Spekulieren, ob das mit the rich get richer zu tun hat – verbinden mit Modell aus Capkun-Paper.

Die grösste Komponente enthält eine erhebliche Anzahl von Knoten mit einem ausgehenden Knotengrad von 1. Ein solch niedriger Grad stellt eine erhebliche Einschränkung für die Benutzbarkeit dieses Schlüssels für die Verifizierung von anderen Schlüsseln dar. Da ein Schlüssel *A* mit ausgehendem Grad 1 nur einen Schlüssel *B* signiert hat, gibt es auch nur genau einen Schlüssel, über den eine Signaturkette aufgebaut werden kann. Natürlich wird dadurch die Menge insgesamt erreichbarer Knoten stark eingeschränkt. Außerdem aber existiert damit keine Redundanz. Der Besitzer des Schlüssels *A* muss dem Besitzer des Schlüssels *B* vertrauen, wenn er irgend eine Signaturkette benutzen will. *B* stellt einen *Single point of failure* dar, da *A* vom Netz getrennt wird, wenn *B* etwa abläuft oder zurückgezogen wird. Äquivalent ist die Verifizierbarkeit eines Knotens *A* stark eingeschränkt, wenn er einen eingehenden Grad von 1 hat und nur von einem einzigen Schlüssel *B* signiert wurde. Für die Verifizierung von *A* muss zwingend dem Besitzer von *B* vertraut werden. Ist *B* nicht mehr benutzbar, wird *A* vom Netz getrennt.

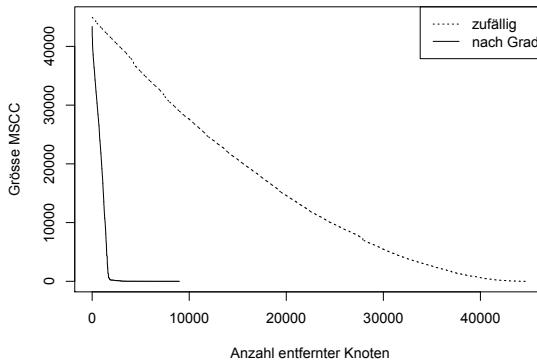


Abbildung 3.4: Entwicklung der Grösse der grössten starken Zusammenhangskomponente, wenn Knoten zufällig oder nach absteigender Grösse entfernt werden

3.1.3 Robustheit

In diesem Zusammenhang stellt sich auch die Frage nach der *Robustheit* des Netzwerkes. Einführung mit Verweis auf Scale-Free. [zufällige Knoten kein Problem, high-deg schon].

Dieser Unterschied kann aufgefasst werden als der Unterschied zwischen einem gerichteten *Angriff* und einer zufälligen Schädigung. Ein Angreifer mit der Intention, das Netz an sich zu schädigen, wird solche Knoten auswählen, deren Entfernung einen möglichst grossen Effekt hat. Demgegenüber kann angenommen werden, dass die Wahrscheinlichkeit einer zufälligen Schädigung in etwa gleichverteilt über alle Schlüssel ist. Ein Angreifer könnte beispielsweise versuchen, Knoten zu entfernen, indem er den Schlüssel an sich kompromittiert oder den Zugriff auf das Schlüsselmaterial unmöglich macht, so dass der Schlüssel widerrufen werden muss. Demgegenüber können Schlüssel auf “normalem” Wege aus dem Netz verschwinden, wenn sie ablaufen oder beispielsweise aufgrund des Verlusts der Passphrase widerrufen werden müssen.

Eine offensichtliche Metrik für die Wichtigkeit eines Knotens und damit eine Strategie für die Auswahl des Zielknotens ist der Knotengrad. Je höher der Grad eines Knotens, desto wichtiger ist er potentiell für das Netzwerk. Um die Robustheit unter dieser Auswahlstrategie zu testen, wurden aus der grössten starken Zusammenhangskomponente sukzessive Knoten in der Reihenfolge absteigender ausgehender Grade bzw. in einer zufälligen Reihenfolge entfernt. Abbildung 3.4 zeigt die Entwicklung der Grösse der grössten Komponente, wenn mehr und mehr Knoten entfernt werden. Bei der Entfernung zufällig ausgewählter Knoten ist das Netzwerk ausgesprochen stabil. Die grösste Komponente schrumpft fast im gleichen Masse, wie Knoten entfernt werden. Dieses Ergebnis ist nicht überraschend. Da ein hoher Anteil der Knoten einen sehr niedrigen Grad hat, ist bei zufälliger Auswahl die Chance, einen solchen Knoten zu treffen, hoch. Das Verschwinden von Schlüsseln durch Ablauen ist ein normaler und häufiger Prozess. So verfügen etwa 5700 Schlüssel in der grössten starken Zusammenhangskomponente über ein Ablaufdatum, werden also irgendwann aus dem Netz verschwinden. Dieser Fall richtet also kaum globalen Schaden im Netzwerk an.

Der Schaden durch die Entfernung von Knoten mit hohem ausgehendem Grad ist deutlich grösser. Hier führt schon die Entfernung von deutlich weniger Knoten zu

einem raschen Verfall bis hin zur völligen Auflösung des Netzwerkes. Aber auch hier richtet die Entfernung *einzelner* Knoten noch keinen signifikanten Schaden an. Erst ab etwa 100 Knoten – was der Entfernung aller Knoten mit einem ausgehenden Grad grösser 250 entspricht – sinkt die Grösse der Komponente erheblich. Die Möglichkeiten eines Angreifers dürften sich eher im Entfernen einzelner Knoten erschöpfen, so dass der zu erwartende Schaden gering ist.

Angemerkt werden muss, dass neben der Grösse der Komponente auch eine Be- trachtung der Entwicklung der Distanzen in der Komponente sinnvoll wäre. Dass die Komponentengrösse sich im Fall zufälliger Reihenfolge stabil verhält, muss nicht bedeuten, dass sich die durchschnittlichen Distanzen nicht deutlich vergrössern. Ver- grössern sich die Distanzen und verlängern sich damit die nötigen Signaturketten, so sinkt die Nützlichkeit des Netzwerkes (siehe Abschnitt 3.1.4). Aufgrund des not- wendigen Rechenaufwandes konnte dies im Rahmen dieser Arbeit aber nicht durch- geführt werden.

Der Grad eines Knoten ist nicht unbedingt das geeignetste Mass für die globale “Wichtigkeit” des Knotens. Durch eine Konzentration auf *zentrale* Knoten, also et- wa solche mit einer hohen Betweenness Centrality, lässt sich möglicherweise mehr Schaden anrichten. Dies lässt sich am Beispiel von Certificate Authorities im Web of Trust illustrieren: CA-Schlüssel haben zwar üblicherweise einen hohen ausgehenden Grad, haben also viele Schlüssel signiert und damit relativ kurze Wege zu vielen Schlüsseln. Allerdings haben sie gleichzeitig einen niedrigen eingehenden Grad, weil ein CA-Schlüssel üblicherweise nicht mit einer Person direkt verbunden ist, und damit das übliche Verifizieren der Identität wenig Sinn macht. Damit tauchen CA- Schlüssel nur auf relativ wenigen kürzesten Pfaden auf. Ihre Entfernung richtet also – zumindest in Bezug auf die Distanzen – einen relativ geringen Schaden an.

3.1.4 Nützlichkeit

Klar ist, dass in einer starken Zusammenhangskomponente jeder Knoten jeden an- deren Knoten erreichen kann. Eine Mindestvoraussetzung für die Verfizierung eines Schlüssels ist, dass eine Signaturkette zu diesem Schlüssel aufgebaut werden kann, dass also der Knoten im Graph erreicht werden kann. Allerdings ist die Erreich- barkeit an sich hier nicht von Bedeutung. Vielmehr muss die Erreichbarkeit unter den Beschränkungen des PGP/GnuPG-Vertrauensmodells (siehe Abschnitt 1.2.3) betrachtet werden. Hier gilt insbesondere die Einschränkung, dass Signaturketten maximal die Länge 5 haben dürfen. Es stellt sich hier die Frage nach der Nützlichkeit des Netzwerkes für dessen eigentlichen Zweck, die Verifizierung von Schlüsseln. Diese ist dann hoch, wenn für viele Knoten eine hohe Zahl von Knoten verifizierbar ist. Selbstverständlich bedeutet das Vorhandensein einer Signaturkette von hinreichend geringer Länge noch nicht, dass ein Schlüssel tatsächlich verifizierbar ist. Zusätzlich muss noch jedem Glied dieser Kette *vertraut* werden. Wird Personen nur *geringfügig* vertraut, sind zusätzlich noch weitere redundante Signaturen nötig. Da diese Ver- trauensinformationen aber nicht öffentlich verfügbar sind, wird hier als obere Grenze für die Anzahl verifizierbarer Schlüssel nur die maximale Pfadlänge verwendet.

Aus Abbildung 3.5b ist ersichtlich, dass die Eccentricity, also die *maximale* Distanz zu irgendeinem Knoten, zwischen dem Minimum 16 (dem *Radius* des Graphen) und dem Maximum 36 (dem *Durchmesser* des Graphen) liegt. Für die meisten Knoten liegt sie bei 26 bis 31. Damit wird die maximale Pfadlänge deutlich überschritten.

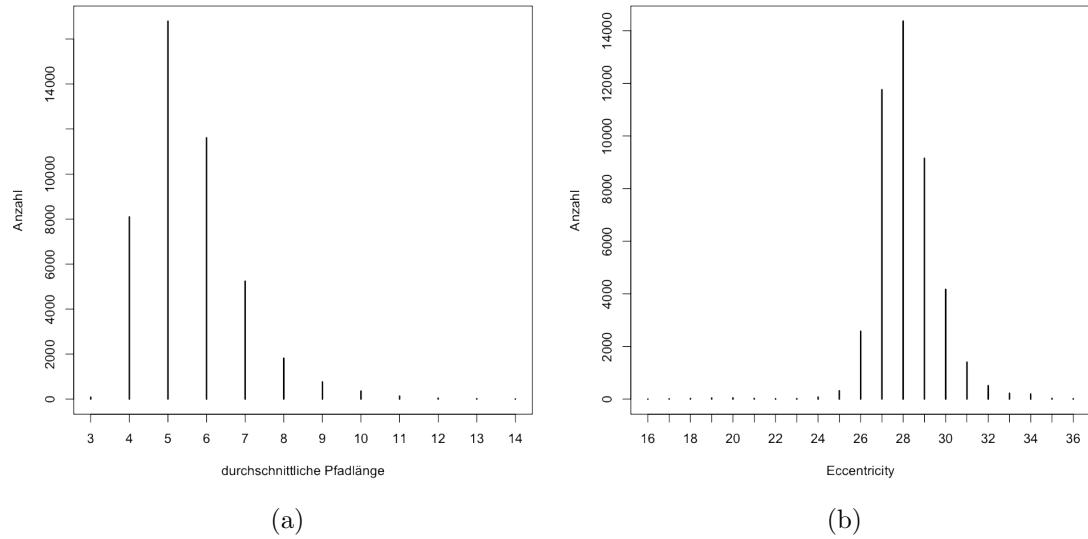


Abbildung 3.5: Verteilung der durchschnittlichen Distanzen (a) und der Eccentricity (b) in der grössten starken Zusammenhangskomponente

Für alle Knoten gilt also, dass die am weitesten entfernten Knoten grundsätzlich nicht erreichbar sind. Aber auch schon die *durchschnittliche* Distanz (Abbildung 3.5) ist für eine erhebliche Anzahl von Knoten grösser als die maximal erlaubte Länge. Klar ist also, dass für die Mehrzahl der Knoten längst nicht alle Knoten unter der Beschränkung der Pfadlänge erreichbar sind.

Die Grösse der 5-Nachbarschaft eines Knotens gibt die Anzahl der Knoten an, die unter dieser Beschränkung erreichbar sind. Allerdings macht es Sinn, nicht nur die 5-Nachbarschaft, sondern auch die n -Nachbarschaften für $n = 2, \dots, 4$, also kürzere Signaturketten, zu betrachten. Da jedem einzelnen Glied einer Signaturkette vertraut werden muss, steigt mit der Länge der Kette auch die Wahrscheinlichkeit, dass dieses Vertrauen eben nicht in alle Glieder vorhanden ist. Da das Web of Trust ein soziales Netzwerk darstellt, kann angenommen werden, dass im Allgemeinen mit der Entfernung zu einem Knoten im Graph auch die soziale und geographische Entfernung zu der entsprechenden Person steigt. Damit sinkt auch die Wahrscheinlichkeit, dass weiter entfernten Personen aufgrund von persönlicher Bekanntschaft vertraut werden kann. Dass einer Person, deren Schlüssel selbst signiert wurde, vertraut wird, ist wahrscheinlicher als dass einer Person vertraut wird, deren Schlüssel nicht selbst signiert wurde. Je länger also eine Signaturkette, desto niedriger die Chance, dass sie tatsächlich benutzbar ist.

Abbildung 3.6 zeigt die kumulative Verteilungsfunktion der Grösse dieser Nachbarschaften. Auffällig ist hier zunächst das steile Wachstum der Kurve für die 2-Nachbarschaft, woraus geschlossen werden kann, dass diese Nachbarschaft für fast alle Knoten sehr klein ist. In der Tat liegt das 2. Quantil bei 30 und das 3. Quantil bei 160. Für die Mehrzahl der Schlüssel ist also die Menge der Schlüssel, für deren Verifizierung nur einer Person voll vertraut werden muss, sehr gering. Immerhin steigt die Grösse der Nachbarschaften mit wachsender Distanz deutlich: Für $n = 3$ liegt das 3. Quantil bereits bei 3371. Für $n = 4$ und $n = 5$ beträgt es 16340 bzw. 30530. Mittels längerer Signaturketten kann also die Mehrzahl der Teilnehmer einen

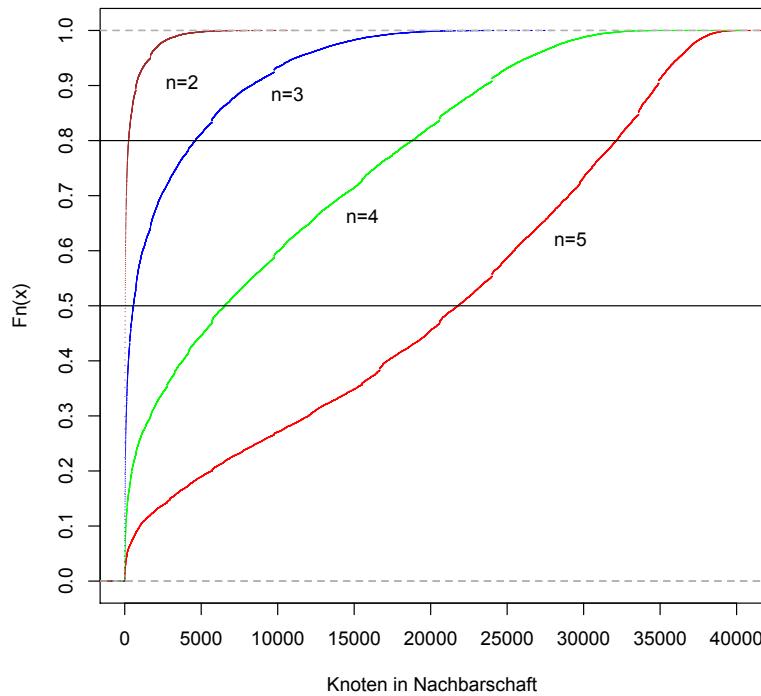


Abbildung 3.6: Kumulative Verteilungsfunktion der Nachbarschaftsgrößen in der grössten starken Zusammenhangskomponente

signifikanten Anteil der in der grössten Zusammenhangskomponente vorhandenen Schlüssel *potentiell* erreichen. Interessanterweise liegt das Maximum der Nachbarschaftsgrößen für $n = 5$ bei nur 40180. Nicht einmal für die am besten vernetzten Schlüssel sind also alle Schlüssel mit beschränkter Kettenlänge erreichbar.

Wie bereits erwähnt, enthält die grösste Zusammenhangskomponente die Schlüssel mehrerer zentraler Certificate Authorities. Interessant ist eine Abschätzung, wie sehr das eigentlich dezentrale Web of Trust auf diese zentralen Komponenten aufbaut. Um dies festzustellen wurden die Schlüssel der grösseren Certificate Authorities (siehe Abschnitt 1.2.4) aus der grössten Komponente entfernt. Diese CA-Schlüssel haben zusammen in dieser Komponente 4256 Schlüssel signiert. Nach der Entfernung der CA-Schlüssel zerfiel die grösste Komponente in eine Reihe von starken Zusammenhangskomponenten: Eine grösste Komponente mit 42455 Schlüsseln und 1058 sehr kleinen Komponenten. Daraus ergibt sich, dass die CA-Schlüssel die grösste Komponente zwar nicht fundamental zusammenhalten. Für immerhin 2497 Schlüssel in der grössten Zusammenhangskomponente ist aber das Vorhandensein und die Benutzung der CA-Schlüssel entscheidend, da sie andernfalls nicht erreichbar sind.

3.2 Eigenschaften einzelner Schlüssel, zeitliche Entwicklung

3.2.1 Zeitliche Entwicklung

Da die verwendete Datenbank alle jemals auf einen Keyserver geladenen Schlüssel enthält es möglich, den Zustand des Schlüsselbestandes zu einem beliebigen Zeit-

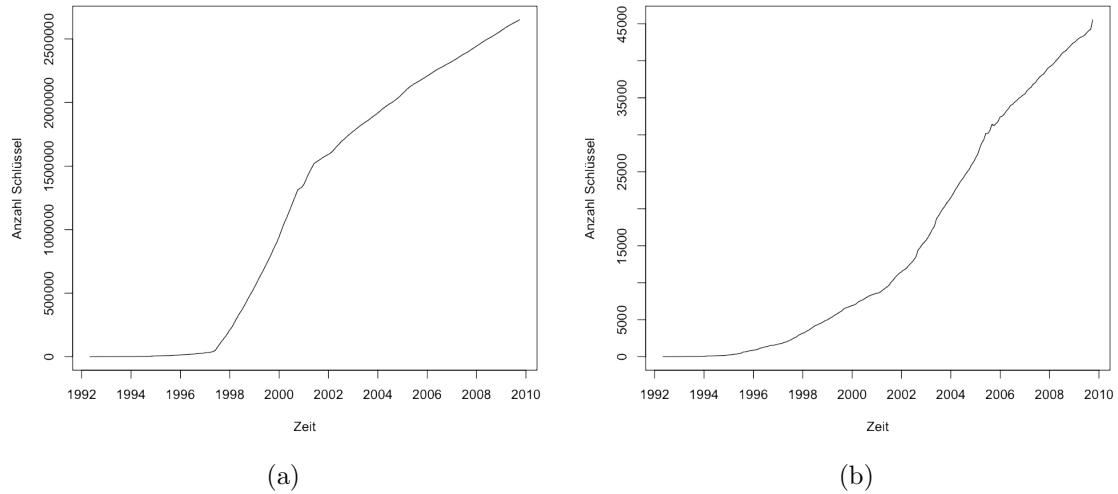


Abbildung 3.7: Zeitliche Entwicklung der Grösse des gesamten Schlüsselbestandes (a) und der grössten starken Zusammenhangskomponente (b)

punkt zu berechnen. In Abbildung 3.7a wird die Entwicklung der Grösse des gesamten Schlüsselbestandes und in 3.7b die Entwicklung der Grösse der grössten starken Zusammenhangskomponente dargestellt. Als Beginn wurde das Jahr 1991 gewählt, weil dort die erste Version von PGP vorgestellt wurde[Wiki10]. Sowohl der gesamte Schlüsselbestand als auch die grösste starke Zusammenhangskomponente zeigen nennenswertes Wachstum erst etwa ab dem Jahr 1997. Dieser Zeitpunkt korreliert unter anderem mit der Gründung des Unternehmens *PGP Inc.*, der Veröffentlichung einer neuen Version 5 der PGP-Software und in Deutschland dem Start der bereits erwähnten “Krypto-Kampagne”. Gründe für das deutlich langsamere Wachstum des gesamten Schlüsselbestandes etwa ab dem Jahr 2001 und der grössten starken Zusammenhangskomponente mit einer Verzögerung etwa ab dem Jahr 2005 sind nicht bekannt.

Abbildung 3.8 zeigt die Rate neuer Schlüssel, abhängig von dem verwendeten Public-Key-Algorithmus. Die DSA-Schlüssel machen zu jedem Zeitpunkt den grössten Anteil aus. Ihre Entwicklung spiegelt im Wesentlichen die Grössenentwicklung insgesamt aus Abbildung 3.8 wieder. Demgegenüber ist die Zuwachsrate von RSA-Schlüsseln bis zum Jahr 2008 konstant niedrig. Der Grund dafür ist vermutlich, dass die PGP-Implementierungen in diesem Zeitraum in der Standardeinstellung DSA-Schlüssel erzeugten, was wiederum unter anderem in dem bis ins Jahr 2000 bestehenden Patentschutz des RSA-Algorithmus begründet ist. Die Rate neuer DSA-Schlüssel sinkt ab dem Jahr 2009 deutlich ab, während die Rate neuer RSA-Schlüssel gleichzeitig deutlich ansteigt. Vermutlich ist dies darauf zurückzuführen, dass seit dem Jahr 2009 neue Schlüssel, die mit GnuPG erzeugt werden, in der Standardeinstellung RSA statt DSA verwenden. Diese Umstellung wurde in der Entwicklerversion von GnuPG am 17.05.2009¹ aufgrund von Berichten über verbesserte Angriffe auf die Hashfunktion SHA-1[McHP09], auf der DSA basiert, vorgenommen. Warum diese Entwicklung nicht ebenfalls in der grössten starken Zusammenhangskomponente sichtbar ist, ist nicht klar. Möglicherweise liegt dies daran, dass für die Aufnahme in diese Teilmen-

¹<http://lists.gnupg.org/pipermail/gnupg-devel/2009-May/025079.html>

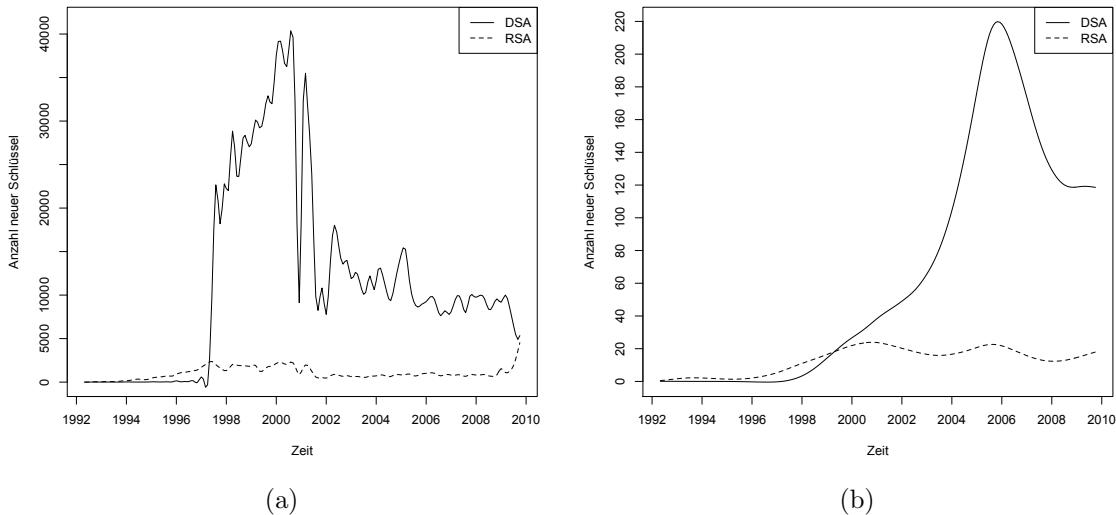


Abbildung 3.8: Rate neuer PGP-Schlüssel, die RSA oder DSA benutzen, für den gesamten Schlüsselbestand (a) und die grösste starke Zusammenhangskomponente (b) in monatlichen Abständen

ge nicht nur das Erzeugen eines Schlüssels, sondern auch zusätzliche Aktivität in Form der Vernetzung mit anderen Schlüsseln notwendig ist. Neue Schlüssel werden also vermutlich im Allgemeinen nicht sofort in die Zusammenhangskomponente aufgenommen, sondern erst mit einiger Verzögerung. Die Zusammenhangskomponente dürfte also etwas “träger” auf Veränderungen reagieren.

3.2.2 Public-Key- und Hashalgorithmen

[SSAL⁺⁰⁹]

Es ist zu erwarten, dass sich der Anteil von RSA-Schlüsseln mit einer Länge von mindestens 2048 bit in nächster Zeit deutlich erhöht.

3.3 Zusammenhangskomponenten und Communities

3.3.1 Communities

Die Zerlegung des gerichteten Graphen mit dem Algorithmus von Rosval et al. lieferte keine verwertbaren Ergebnisse. Zwar wurde eine Zerlegung in 2869 Partitionen berechnet. Allerdings gilt für die grosse Mehrzahl dieser Knotenmengen, dass der durch sie induzierte Teilgraph überhaupt keine Kanten hat. Nur für ca. 160 Partitionen hat der jeweils induzierte Teilgraph Kanten, in allen Fällen allerdings sehr wenige (zwischen 1 und 104 für eine Partition der Grösse 682). Dass die Knoten einer Partition in allen Fällen so gut wie keine interne Vernetzung haben und damit der Definition von Communities absolut nicht entsprechen, ist ein klarer Hinweis, dass die berechnete Zerlegung die tatsächliche modulare Struktur des Graphen nicht sinnvoll wiedergibt. Dass der Graph in der Tat eine ausgeprägte modulare Struktur hat, zeigen die anhand des ungerichteten Graphen berechneten Zerlegungen, die

auch unter inhaltlichen Gesichtspunkten Sinn ergeben. Diese Struktur kann sich auch nicht erst durch die Reduzierung des gerichteten auf einen ungerichteten Graphen ergeben. Die berechnete Zerlegung mit etlichen grossen Partitionen ohne interne Vernetzung ist in keinem Fall ein sinnvolles Ergebnis eines Community-Algorithmus und spricht daher für eine fehlerhafte Berechnung. Ob diese allerdings auf Fehler in der verwendeten Implementierung der Autoren² oder auf Fehler im Algorithmus zurückgeht, ist nicht bekannt.

Für die restlichen Methoden wurde der Graph auf einen ungerichteten Graphen reduziert. Dazu wurde jede einzelne gerichtete Kante in eine ungerichtete Kante umgewandelt. Aus dem gerichteten Graphen mit 446326 Kanten ergab sich so ein ungerichteter Graph mit 295425 Kanten.

Für die COPRA-Berechnung wurden – wie in [Greg10] vorgeschlagen – Berechnungen mit unterschiedlichen Werten v durchgeführt, um den “besten” Wert im Sinne der Modularity zu ermitteln. Dabei ergaben sich mit steigender maximaler Überlappung v nur bis $v \leq 3$ steigende Modularity-Werte, darüber hinaus fielen sie kontinuierlich ab. Deshalb wurde die Berechnung mit $v = 3$ verwendet. Die dort berechneten Modularity-Werte hatten eine geringe Standardabweichung von 0,003. Dieses Ergebnis kommt unerwartet. In [Greg10] ergaben sich für ein PGP-Web-of-Trust-Netzwerk mit 10680 Knoten bis zu $v = 9$ ansteigende Modularity-Werte. Das (lokale) Maximum bei geringem v für das hier verwendete Netzwerk könnte darauf hinweisen, dass seine Community-Struktur keine signifikante Überlappung aufweist. Ein weiterer Hinweis darauf ist, dass die Einführung der Überlappung mit $v = 3$ gegenüber der nicht überlappenden Berechnung mit $v = 1$ für die Modularity (Tabelle 3.1) keinen signifikanten Anstieg brachte. Auch die Zuordnung zu Gruppen bzw. Keysigning-Parties (Tabelle 3.2) und die Verteilung der Grösse der Communities (Abbildungen 3.9b und 3.9a) ähneln sich weitgehend. Alternativ könnte dieses Ergebnis allerdings auch in einem nicht-optimalen Verhalten des Algorithmus begründet liegen.

Für den Algorithmus von Blondel et al. zeigt sich zunächst ein deutlich höherer Modularity-Wert als für Fastmod. Dieses Ergebnis gibt also die reine Struktur des Graphen deutlich besser wieder. Wie in Abschnitt 1.5.2 beschrieben, verläuft der Algorithmus von Blondel et al. iterativ in Phasen, in denen jeweils zuerst Knoten einer benachbarten Community zugeordnet werden, wenn sich dadurch ein Modularity-Gewinn erzielen lässt, und dann die Knoten dieser so entstandenen Communities zu Knoten in einem neuen Netzwerk verschmolzen werden. Hier werden die Ergebnisse der Phasen $l = 2$ und der letzten Phase $l = 5$ betrachtet. Für $l = 5$ ergab sich die höchste Modularity. Allerdings sticht hier die sehr kleine Anzahl von Communities heraus, die wiederum vergleichsweise gross sein müssen. Gegenüber dem Ergebnis der Phase 2 ergibt sich eine relativ kleine Steigerung der Modularity, die mit einem erheblichen Verlust an *Auflösung* erkauft wird³. Aus Abbildung 3.9c und 3.9d geht hervor, dass diese Reduktion der Anzahl im Wesentlichen auf das Reduzieren sehr kleiner Communities zurückgeht. Da der Algorithmus von Blondel et al. genauso wie Fastmod mittels einer (lokalen) Optimierung der Modularity arbeitet, könnte dieses Phänomen auf das in der Literatur beschriebene Auflösungslimit der Modularity-

²<http://www.tp.umu.se/~rosvall/code.html>

³Die Zerlegungen der Phasen 3 und 4 unterschieden sich nur geringfügig von der Zerlegung der Phase 5

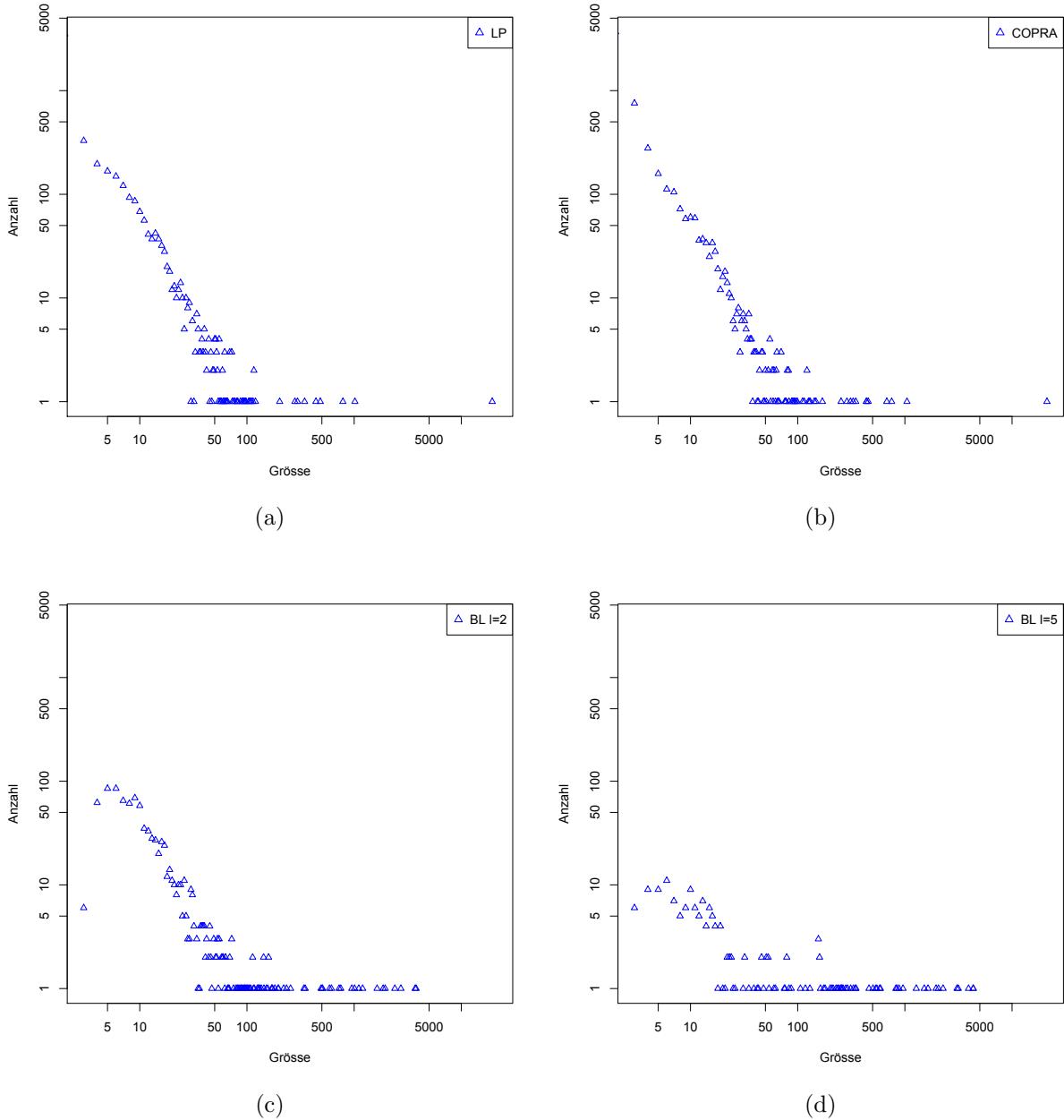


Abbildung 3.9: Größenverteilung der Communities

Funktion (bspw. [FoBa07] [GodC09]) zurückzuführen sein. Es scheint fraglich, ob die in Relation zur Grösse des Graphen sehr geringe Zahl von 186 Communities die Struktur des Graphen (in Bezug auf die inhaltliche Analyse) adäquat wiedergibt.

Abbildung 3.10 illustriert dies anhand einer Community aus der Zerlegung des Algorithmus von Blondel et al. mit $l = 5$, die klar abgegrenzte Untereinheiten enthält, die intuitiv wieder eigenständige Communities darstellen.

Abbildung 3.11 illustriert die begrenzte Qualität der Fast-Modularity-Zerlegung, die sich im vergleichweise niedrigen Modularity-Wert niederschlägt. In der Zeichnung des induzierten Teilgraphen einer grossen Community sind etliche dicht gezeichnete Zusammenballungen von Knoten zu sehen, die nur wenige Kanten nach aussen haben. Diese können als Community-artige Strukturen interpretiert werden. Zwar kann

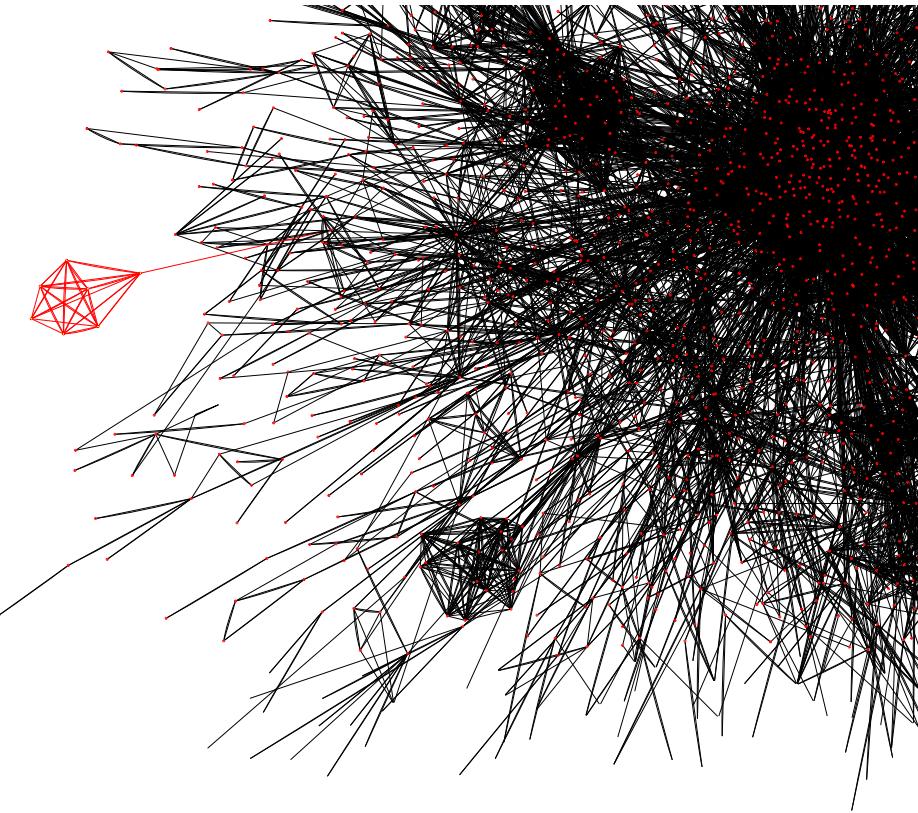


Abbildung 3.10: Community mit modularen Untereinheiten (Grösse 2271, Blondel ($l = 5$), Force-directed Layout)

argumentiert werden, dass die Zeichnung die Struktur des Teilgraphen nicht notwendigerweise sinnvoll wiedergibt, dass also die dichten Teilbereiche in der Zeichnung nur Artefakte der Zeichenmethode darstellen. Allerdings zeigt Noack, dass starke Übereinstimmungen zwischen einer energieoptimalen Einbettung eines Graphen in die Ebene nach der Force-directed-Methode und einer Partitionierung eines Graphen mit optimaler Modularity bestehen [Noac09]. Es scheint daher gerechtfertigt, eine Zeichnung eines Graphen (Force-directed) mit markanten dichten Bereichen als Indiz für eine modulare Struktur dieses Graphen zu betrachten.

Aus Tabelle 3.1 geht hervor, dass sich – zumindest im Fall von Fastmod und Blondel et al. – fast alle Knoten des Graphen in einer Community wiederfinden, die mindestens die Grösse 4 hat. In Abschnitt 3.1 wurde gezeigt, dass ein erheblicher Anteil der Knoten einen sehr geringen Grad von 1 oder 2 hat. Da solche Knoten mangels Kanten nicht selbst eine dichte Vernetzung herstellen können, müssen sie Mitglied in Communities sein, in denen Knoten mit höherem Grad sowohl die interne Vernetzung der Community als auch die Vernetzung der Community mit anderen erzeugen. COPRA scheint hingegen dazu tendieren, deutlich mehr Knoten in Communities sehr kleiner Grösse mit 3 oder weniger unterzubringen. Dieses Verhalten könnte dazu führen, dass die Aussagekraft der Zerlegung in Bezug auf die realen Communities geschwächt wird, wenn viele Knoten auf diese Art abgespalten werden.

Gegenüber der COPRA-Zerlegung mit $v = 1$ zeigt die Zerlegung mit $v = 3$ eine deutlich grössere Anzahl von Communities der Grösse 3 und 4. Abgesehen davon ähneln sich die Grössenverteilungen aber weitgehend. Beide enthalten eine charak-

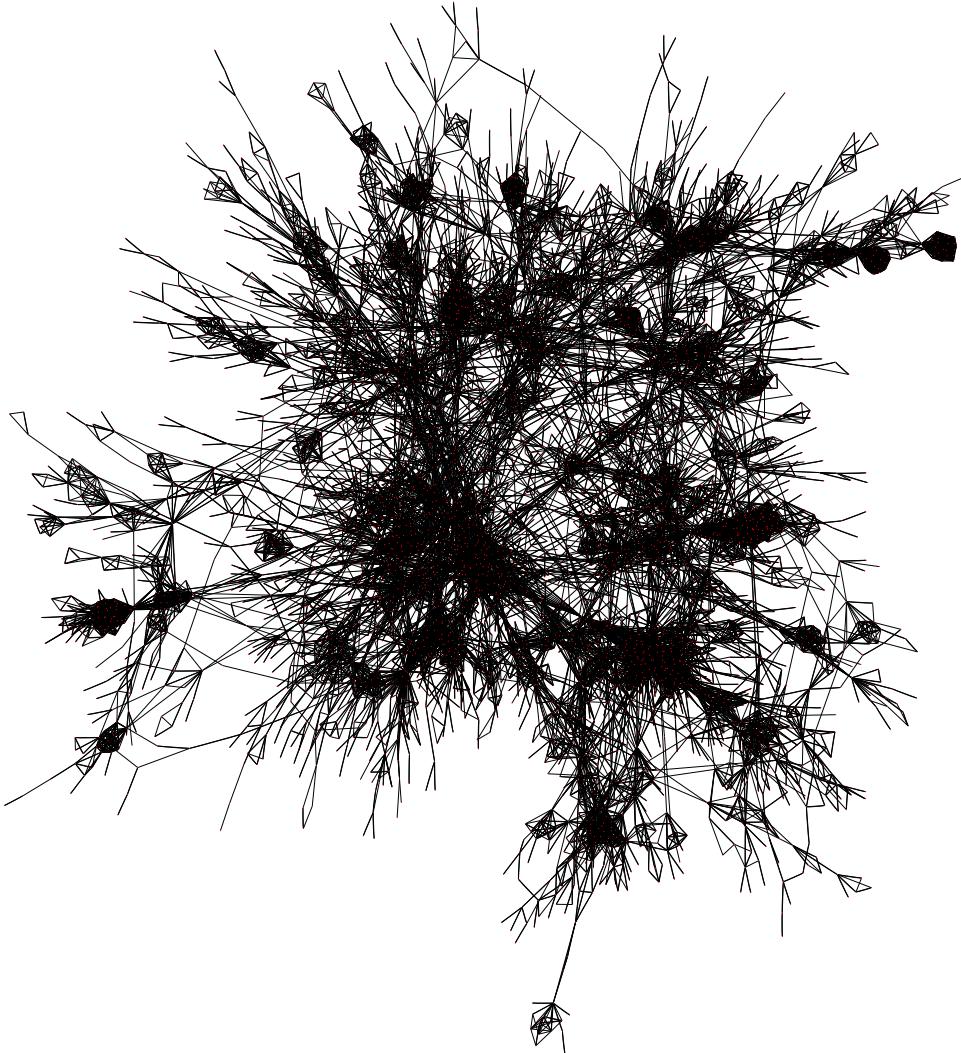


Abbildung 3.11: Grosse Community mit modularer Struktur (Berechnet mit Fast-Modularity, Force-directed layout)

teristische sehr grosse Community mit ca. 19000 bzw. ca. 21000 Mitgliedern. Demgegenüber bietet die Blondel-Zerlegung mit $l = 2$ eine deutlich geringere Anzahl solcher sehr kleiner Communities, dafür aber eine höhrere Zahl mittlerer (mehr als 100 Mitglieder) und insbesondere eine höhere Zahl sehr grosser (zwischen 500 und 5000 Mitglieder) Communities. Die beiden Algorithmen konvergieren nicht gegen ein gemeinsames Optimum, sondern liefern Ergebnisse, die sich qualitativ stark unterscheiden.

Der insgesamt recht hohe Modularity-Wert weist darauf hin, dass der Graph in der Tat eine ausgeprägte modulare Struktur hat, wie dies von einem sozialen Netzwerk erwartet wird.

Die Abbildungen 3.12 und 3.13 zeigen die Struktur der Communities für die Zerlegungen COPRA ($v = 1$) und Blondel ($l = 2$). Auffällig ist hier insbesondere die teilweise sternförmige Struktur der COPRA-Zerlegung. Ein signifikanter Anteil der kleineren Communities ist rund um die zentrale Community der Grösse ca. 19000 angeordnet und im wesentlichen nur mit dieser vernetzt. Diese Struktur weist gewisse Ähnlichkeiten zur Struktur der starken Zusammenhangskomponenten auf (siehe

| Algorithmus | Modularity | Communities | Enthaltene Knoten |
|-------------------|------------|-------------|-------------------|
| FM | 0,596 | 552 | 44611 |
| BL ($l = 2$) | 0,701 | 936 | 44934 |
| BL ($l = 5$) | 0,710 | 186 | 44934 |
| COPRA ($v = 1$) | (0,780) | 1421 | 42205 |
| COPRA ($v = 3$) | (0,786) | 1354 | — |

Tabelle 3.1: Modularity-Werte, Anzahl der Communities mit mehr als 3 Knoten sowie die Anzahl der insgesamt in diesen Communities (>3) enthaltenen Knoten für die Algorithmen Fast-Modularity (FM), Label-Propagation (LP), Blondel et al. (BL) auf Level 2 und 5 sowie COPRA. Die Modularity-Werte für COPRA entsprechen der Modularity-Definition für überlappende Zerlegungen nach [NMCM09] und können mit den anderen Werten nicht verglichen werden.

| Algorithmus | TLD-S | TLD-M | SLD-S | SLD-M | SIG |
|-------------------|-----------|-----------|-----------|-----------|-----------|
| FM | 293 (53%) | 247 (44%) | 56 (10%) | 154 (27%) | 128 (23%) |
| BL ($l = 2$) | 499 (53%) | 417 (45%) | 41 (4%) | 254 (27%) | 115 (12%) |
| BL ($l = 5$) | 85 (47%) | 85 (47%) | 15 (8%) | 38 (21%) | 26 (14%) |
| COPRA ($v = 1$) | 824 (58%) | 564 (40%) | 178 (13%) | 429 (30%) | 572 (40%) |
| COPRA ($v = 3$) | 792 (58%) | 525 (38%) | 187 (14%) | 425 (31%) | 555 (41%) |

Tabelle 3.2: SLD-TLD-Zuweisung, TIME-CORR

Abschnitt ??). Diese sind ebenfalls sternförmig um eine zentrale Komponente angeordnet, die die Grösse der anderen um mehrere Größenordnungen übertrifft. Allerdings sind im Fall der Community-Struktur noch eine Reihe mittelgrosser Communities vertreten, die dieses Muster etwas stören. Das sternförmige Muster der Zusammenhangskomponenten wiederholt sich also in etwa innerhalb der grössten Zusammenhangskomponente auf einem anderen Level. Es könnte daher spekuliert werden, ob das Netzwerk eine *Selbstähnlichkeit* aufweist, wie sie für eine Reihe von komplexen Netzwerken gezeigt wurde[SoHM05]. Dieser Frage wurde allerdings im Rahmen dieser Arbeit nicht weiter nachgegangen. Im Vergleich dazu zeigt sich in der Struktur der Blondel-Communities die etwas ausgewogenere Größenverteilung. Zwar sind auch hier viele kleine Communities sternförmig um die zentralen Communities angeordnet, verteilen sich hier aber auf mehrere solcher grossen zentralen Communities.

Aus Tabelle 3.2 geht hervor, dass für alle Zerlegungen fast alle Communities einer Top-Level-Domain zuordnenbar sind oder von einer TLD dominiert werden. Werden die generischen Top-Level-Domains (gTLD: com, org, net, info, ...) abgezogen, verbleiben im Fall von COPRA noch 519 (38%) Communities, die einer TLD (und damit einer *Country-Coded TLD (ccTLD)*) zugeordnet werden können und 315 (23%), die von einer ccTLD dominiert werden. Für die anderen Zerlegungen ergeben sich ähnliche Werte. Damit ist immerhin etwa die Hälfte der Communities einem Land zuordnenbar. Für die von einer gTLD dominierten Communities ergeben Stichproben, dass sich die Teilnehmer der meisten dieser Communities anhand der Namen zumindest einem Sprachraum zuweisen lassen.

Dieses Ergebnis ist naheliegend. Wie in Abschnitt 2.3 dargelegt, wird davon ausgegangen, dass ein wesentlicher Faktor, der das Zustandekommen von Signaturen beeinflusst, die sozialen Kontakte einer Person sind. Es ist weiterhin naheliegend, dass sich diese sozialen Kontakte unter anderem aufgrund von Sprachbarrieren und

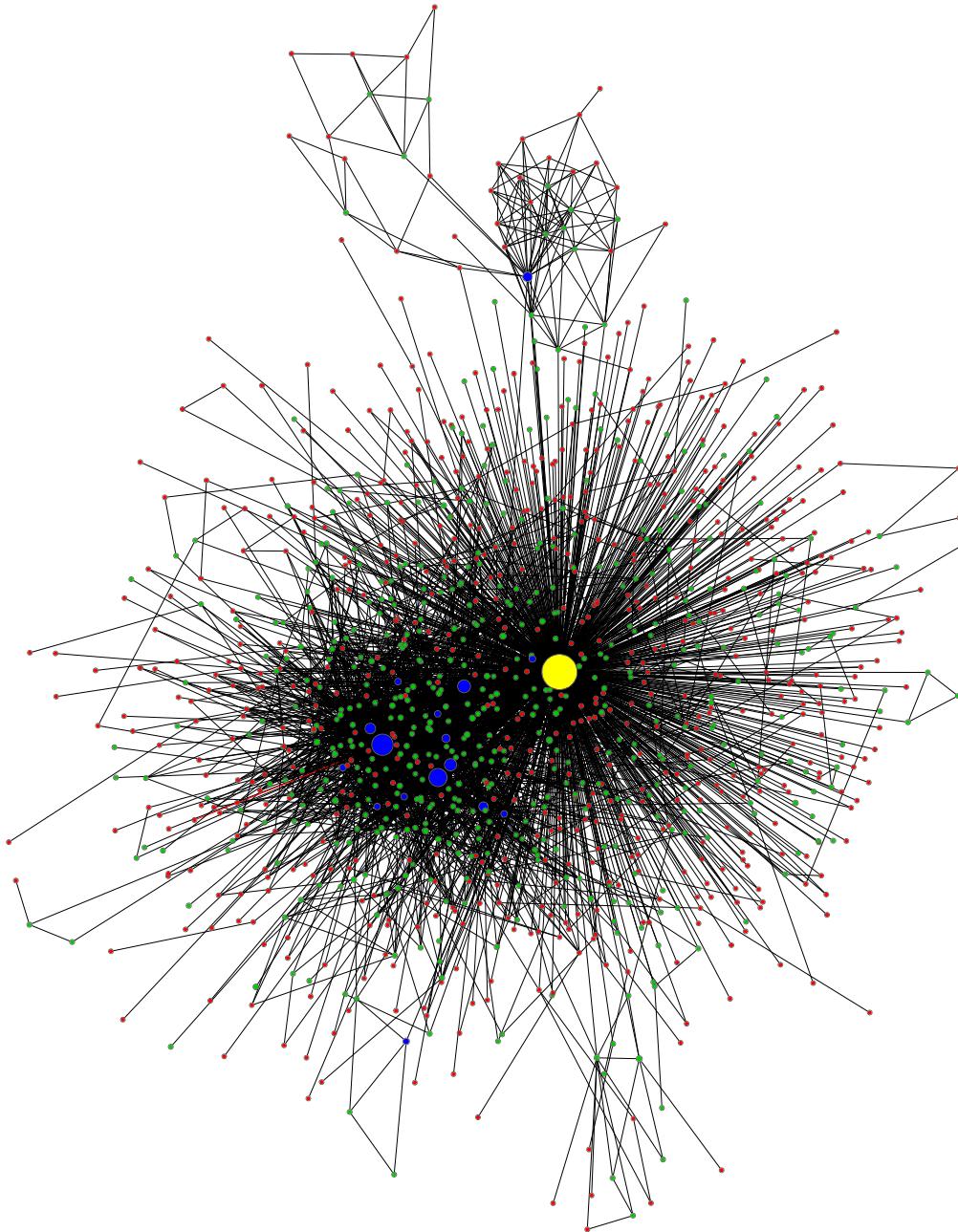


Abbildung 3.12: Struktur der Communities (COPRA ($v = 1$) (rot: Grösse < 10 , grün: Grösse < 100 , blau: Grösse < 1000 , gelb: Grösse > 1000).

der räumlichen Nähe hauptsächlich im Rahmen eines Landes ergeben. Beispielsweise ist für eine aus Finnland stammende Person anzunehmen, dass die Mehrzahl ihrer Freunde, Bekannten, Geschäftspartner und sonstigen Kontakte wieder Finnen sind.

Zwar könnte argumentiert werden, dass das Internet die geographische bzw. nationale Beschränkung sozialer Kontakte aufhebt. Allerdings spielt die geographische Nähe eine wichtige Rolle im Signaturprozess. Die verbreitete Prozedur für das Signieren eines Schlüssels setzt ein direktes persönliches Treffen der Signierenden voraus (siehe Abschnitt 1.2.4). Es ist anzunehmen, dass die überwiegende Mehrheit der PGP-Benutzer die meiste Zeit in einem Land und damit in einem geographisch eingegrenzten Bereich verbleibt. Damit kommen für diese Mehrheit überwiegend

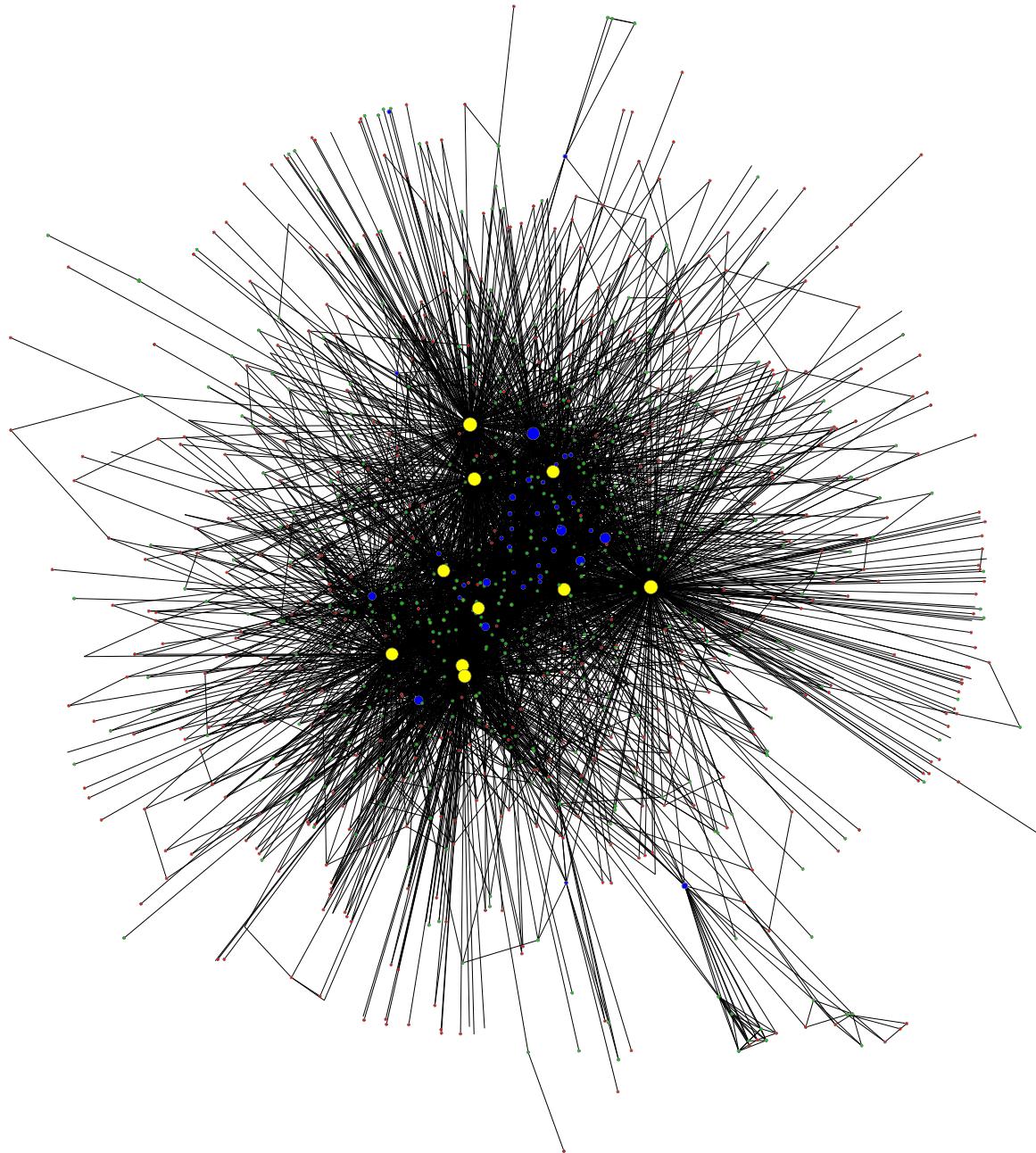


Abbildung 3.13: Struktur der Communities (Blondel ($l = 5$) (Knotenfärbung wie in Abb.3.12)

nur Bewohner des gleichen Landes als Signaturpartner in Frage. Als Mechanismen für die internationale Vernetzung kommen dann etwa internationale Konferenzen in Frage. Nur ein sehr kleiner Teil der PGP-Benutzer dürfte die notwendige internationale Mobilität aufweisen, so dass sich seine Signaturen nicht überwiegend einem geographischen Bereich bzw. einem Land zuweisen lassen.

Interessanter für die Fragestellung ist aber die Erkennung von Keysigning-Parties und die Zuordnung zu Second-Level-Domains. Im Vergleich der Algorithmen (Tabelle 3.2) zeigt sich zunächst, dass die COPRA-Zerlegungen bessere Resultate liefern als die Blondel-Zerlegungen, da hier mehr Communities zugeordnet werden können. Zwar entfällt bei COPRA ein grosser Anteil dieser Zuordnungen auf sehr kleine Communities der Grösse 3 und 4. Die Aussagekraft solcher kleiner Zuordnungen

ist selbstverständlich sehr begrenzt. Werden diese abgezogen, verbleibt aber immer noch eine grösse Zahl zuordnenbarer Communities nichttrivialer Grösse. Aus den absoluten Zahlen in Tabelle 3.2 und den Grössenverteilungen in Abbildungen 3.16, 3.17 und 3.18 geht hervor, dass die Überlappung keine wesentlichen Vorteile bringt. Im Vergleich der Blondel-Zerlegungen zeigt sich, dass das Zusammenpacken von Communities von Phase $l = 2$ zu Phase $l = 5$ und der damit verbundene Auflösungsverlust es erheblich erschwert, die Communities inhaltlich zuzuordnen. Eine Zerlegung mit höherer Modularity entspricht also im Allgemeinen nicht einer Zerlegung, die die Struktur realer Communities besser wiedergibt. Die Zerlegungen des Algorithmus von Blondel et al. scheinen insgesamt vom Auflösungslimit der Modularity-Maximierung betroffen zu sein und bieten daher nicht die notwendige Auflösung, um die Communities besser inhaltlich analysieren zu können.

Insgesamt ist eine tatsächliche Zuordnung anhand des 80%-Schwellwerts nur für sehr wenige Communities von geringer Grösse möglich. Für den 40%-Schwellwert und die zeitliche Korrelation ist die Anzahl zuordnenbarer Communities deutlich höher. Auch hier entfällt allerdings die Mehrzahl auf sehr kleine Communities. Dass so gut wie keine Communities mit einer Grösse über 100 zugeordnet werden können, spricht dafür, dass diese sich nicht anhand dieser einfachen Mechanismen bilden.

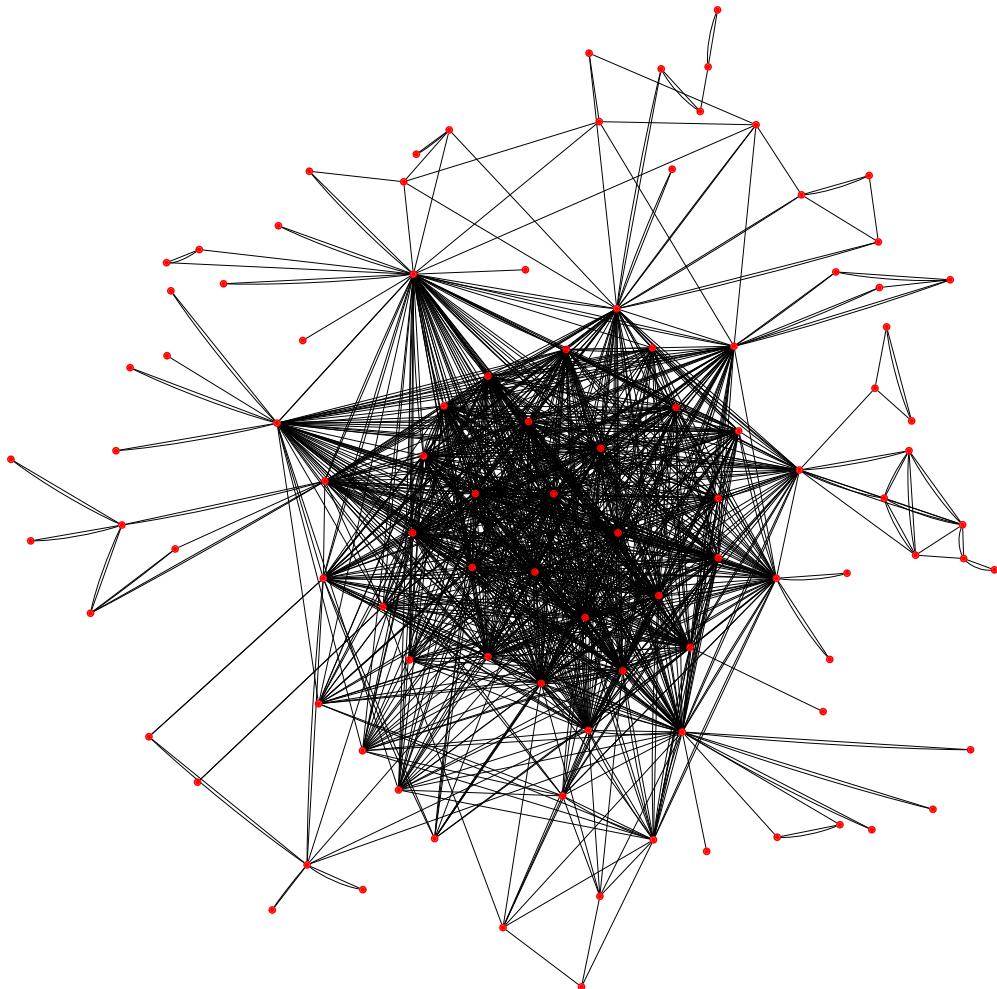
Die sehr grosse Community aus der COPRA-Zerlegung könnte vermuten lassen, dass hier relativ willkürlich Knoten zusammengepackt wurden. Allerdings hat diese Community eine Reihe von Eigenschaften, die nahelegen, dass diese Gruppierung durchaus Sinn macht. Von 21157 Knoten vefügen 8879 (42%) über eine E-Mail-Adresse in der Top-Level-Domain `.de`, während andere – insbesondere nicht-deutschsprachige – ccTLDs deutlich schwächer vertreten sind. Das zeigt einerseits einen überraschend hohen Anteil von deutschen oder deutschsprachigen Benutzern am Gesamtnetzwerk. Die 8879 Schlüssel sind natürlich nur eine Untergrenze, da auch in einer Vielzahl weiterer Communities deutsche Teilnehmer vorkommen. Ein Grund für diese hohe Anzahl deutscher Schlüssel könnte in der in Abschnitt 1.2.4 erwähnten “Krypto-Kampagne” der Zeitschrift c’t liegen, die das Bewusstsein für die Notwendigkeit der Authentifizierung von Schlüsseln im deutschsprachigen Raum deutlich gesteigert haben dürfte. Andererseits lässt dies den Schluss zu, dass die enthaltenen Knoten eben nicht willkürlich zusammengepackt wurden, sondern neben ihrer topographischen Nähe auch eine “inhaltliche” Gemeinsamkeit aufweisen. Darauf weist auch hin, dass in dieser Community eine relativ zu anderen SLDs sehr grosse Anzahl von Schlüsseln von Mitgliedern des Debian-Projekts enthalten ist. Diese Community enthält nicht nur knapp die Hälfte der Schlüssel, sondern mit ca. 260000 Kanten den Grossteil der überhaupt vorhandenen Kanten. Sie ist also anscheinend intern sehr dicht vernetzt. Daher scheint es fraglich, ob sie intern wieder eine schlecht aufgelöste modulare Struktur wie in Abbildung 3.11 hat, die sich in sinnvolle Untereinheiten zerlegen lässt.

In Tabelle 3.3 sind die 6 grössten Communities aus der COPRA-Zerlegung mit Zuordnung zu Second-Level-Domains aufgeführt.

Es stellt sich die Frage, ob die zeitliche Korrelation der Signaturen in einer Community ein geeignetes Mittel ist, um Keysigning-Parties zu erkennen. Neben dem zeitlichen Zusammenhang der Signaturen ist ein weiteres Merkmal einer Keysigning-Party wie in Ab. 1.2.4 beschrieben, dass jeder Teilnehmer mit (fast) jedem anderem Teilnehmer Signaturen austauscht, so dass sich eine fast vollständige Vermaschung –

| Grösse | SLD | Anteil |
|--------|------------|--------|
| 436 | apache.org | 48% |
| 130 | nun.org | 82% |
| 97 | cert.org | 70% |
| 81 | redhat.org | 68% |
| 63 | purdue.edu | 65% |
| 59 | cisco.com | 79% |

Tabelle 3.3: foo

Abbildung 3.14: Community mit zeitlicher Korrelation der Signaturen (Blondel ($l = 5$), 100 Knoten, 72% der Signaturen innerhalb eines Monats)

fast eine Clique – ergibt. Eine stichprobenartige Untersuchung in den Communities zeigt, dass die meisten Communities mit zeitlicher Korrelation zumindest teilweise dieses Merkmal zeigen. Abbildung 3.14 zeigt ein Beispiel einer solchen Community. Etwa die Hälfte der Knoten ist in einer Fast-Clique enthalten, deren Knoten mit (fast) allen anderen vernetzt sind. Dieser innere Teil des Teilgraphen entspricht genau dem Bild, das als Resultat einer Keysigning-Party erwartet wird. Auch wenn dies auf die meisten der Communities mit zeitlicher Korrelation zutrifft, gibt es auch Gegenbeispiele. Abbildung 3.15 zeigt eine Community, die zwar das Kriterium der zeitlichen Korrelation erfüllt, ansonsten aber nichts mit dem erwarteten Bild einer

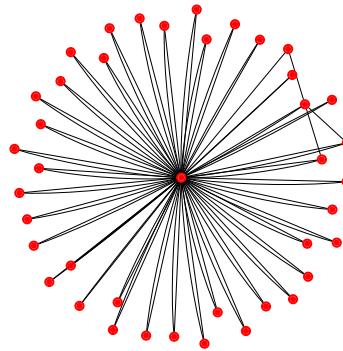


Abbildung 3.15: Community mit zeitlicher Korrelation der Signaturen (Blondel ($l = 5$), 41 Knoten, 84 % der Signaturen innerhalb eines Monats)

KSP gemein hat. Allerdings erfüllt der Teilgraph trotzdem das Kriterium der Community, da die Knoten am Rand nur Signaturen zu dem zentralen Knoten haben und damit die interne Kantendichte deutlich höher ist als die externe. Um solche Extremfälle auszuschliessen, ist als zusätzliches Kriterium für die Erkennung von KSPs ein hoher durchschnittlicher Grad der Knoten denkbar.

Diese Ergebnisse widerlegen nicht die Annahme, dass die Vernetzung im Web of Trust im wesentlichen von Keysigning-Parties und den sozialen Kontakten der Teilnehmer beeinflusst wird und damit unter anderem von Gruppenzugehörigkeit bestimmt wird. Allerdings muss festgehalten werden, dass die hier verwendeten Methoden nicht geeignet sind, um den Entstehungsmechanismus entlang dieser Annahme befriedigend zu erklären.

Ein Grund dafür ist sicherlich, dass die Struktur des Web of Trust nur zu einem bestimmten, festen Zeitpunkt betrachtet wurde. Um die Entstehungsdynamik zu verstehen, könnte alternativ die Entwicklung des Netzwerks tatsächlich über die Zeit nachvollzogen werden. Das Keyserver-Netzwerk speichert nicht nur den momentanen Zustand des Netzwerks, sondern implizit auch den Entstehungszeitpunkt jedes Schlüssels und jeder Signatur. Die im Rahmen dieser Arbeit entwickelte Software speichert diese Zeitstempel in einer Datenbank und macht so die gesamte Entwicklungsgeschichte des Web of Trust verfügbar (siehe Abschnitt 2.2.2). Anhand dieser Daten könnte nachvollzogen werden, wie und in welcher Form die Communities tatsächlich entstehen und sich entwickeln.

Dass nur sehr wenige grössere Communities mit zeitlicher Korrelation der Signaturen gefunden wurden, obwohl regelmässig grössere Keysigning-Parties stattfinden (siehe Abschnitt 1.2.4) ist bei näherer Betrachtung durchaus plausibel. Dass eine Keysigning-Party sich in der Struktur markant abhebt, setzt voraus, dass ihre Teilnehmer ausser der Teilnahme an dieser Party keine wesentliche Signaturaktivität starten. Würden sie weiterhin an Signaturaktivitäten teilnehmen, würde die Struktur der Keysigning-Party mit der Zeit durch Signaturen von und zu Nicht-Teilnehmern “verwischt” werden. Aufgefunden wurden also vermutlich nur die Keysigning-Parties mit einer Mehrheit von insgesamt eher inaktiven Teilnehmern. Gleiches kann für Teilnehmer angenommen werden, die sich anhand von Gruppenzugehörigkeit vernetzen. Auch hier würde die Struktur mit der Zeit “unschärfer” werden, wenn ihre Mitglieder weiterhin Signierungen vornehmen würden. Es kann also angenommen werden, dass die kleineren und insbesondere die klar zuordnenbaren Communities

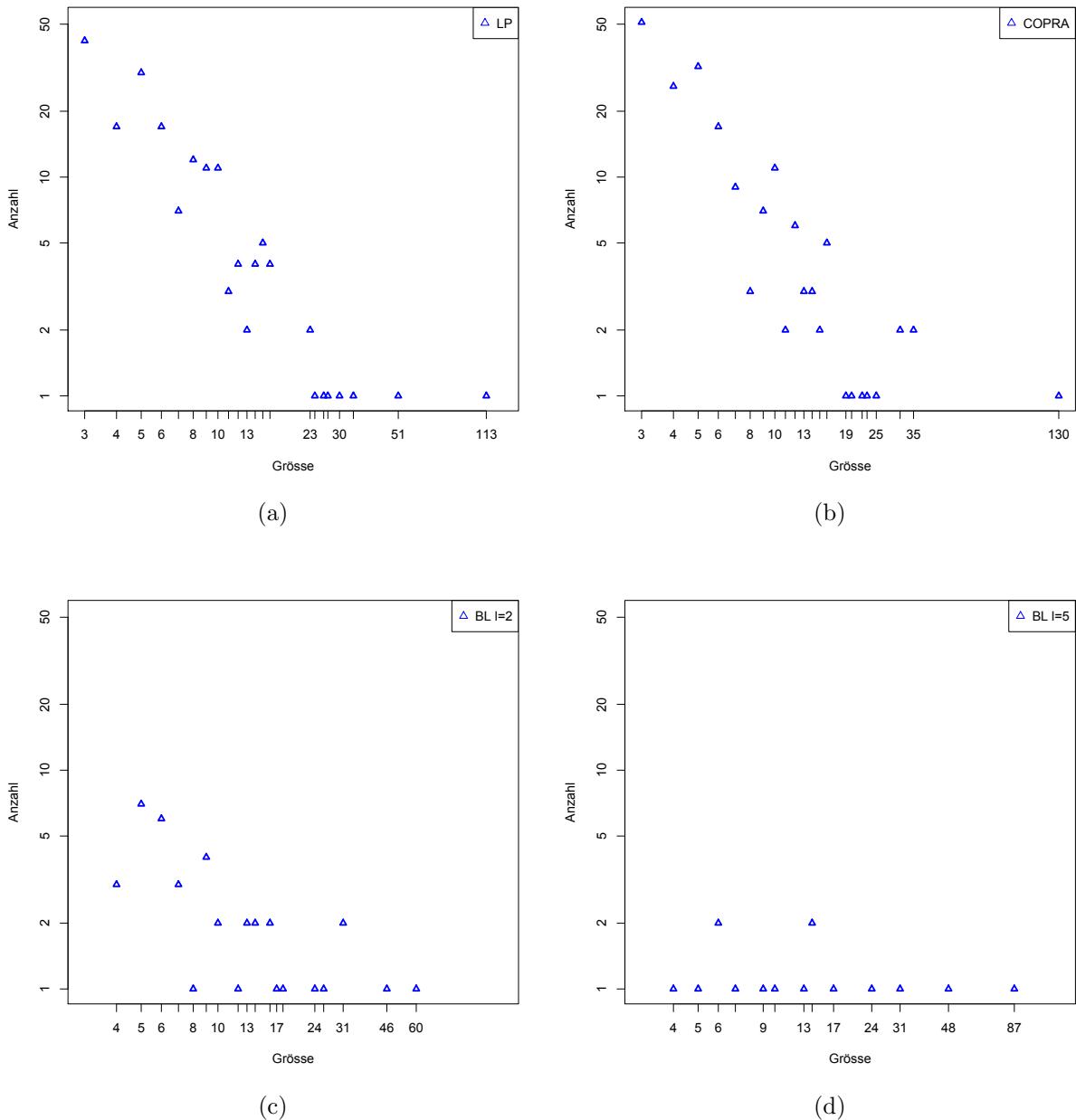


Abbildung 3.16: Zuweisung von Domains zu SLDs abhängig von der Community-Grösse

tendenziell Teilnehmer enthalten, die in Bezug auf die Teilnahme am Web of Trust eher inaktiv sind. Um diese Vermutung zu bestätigen oder zu widerlegen, könnte das Signaturverhalten der Teilnehmer über die Zeit mit ihrer Community-Mitgliedschaft verglichen und korreliert werden.

Es scheint unrealistisch, dass die Mitgliedschaft in einer Gruppe – etwa einer Firma, akademischen Einrichtung oder einem Open-Source-Projekt – die sozialen Kontakte einer Person vollständig charakterisiert. Daneben kann eine Person noch ein weites Netzwerk an Freunden oder Bekannten haben, die diese Mitgliedschaft nicht teilen. Trotzdem können sich zu ihnen Signaturen ergeben, die einen wesentlichen Teil der Signaturen dieser Person ausmachen können. Außerdem ist anzunehmen, dass eine

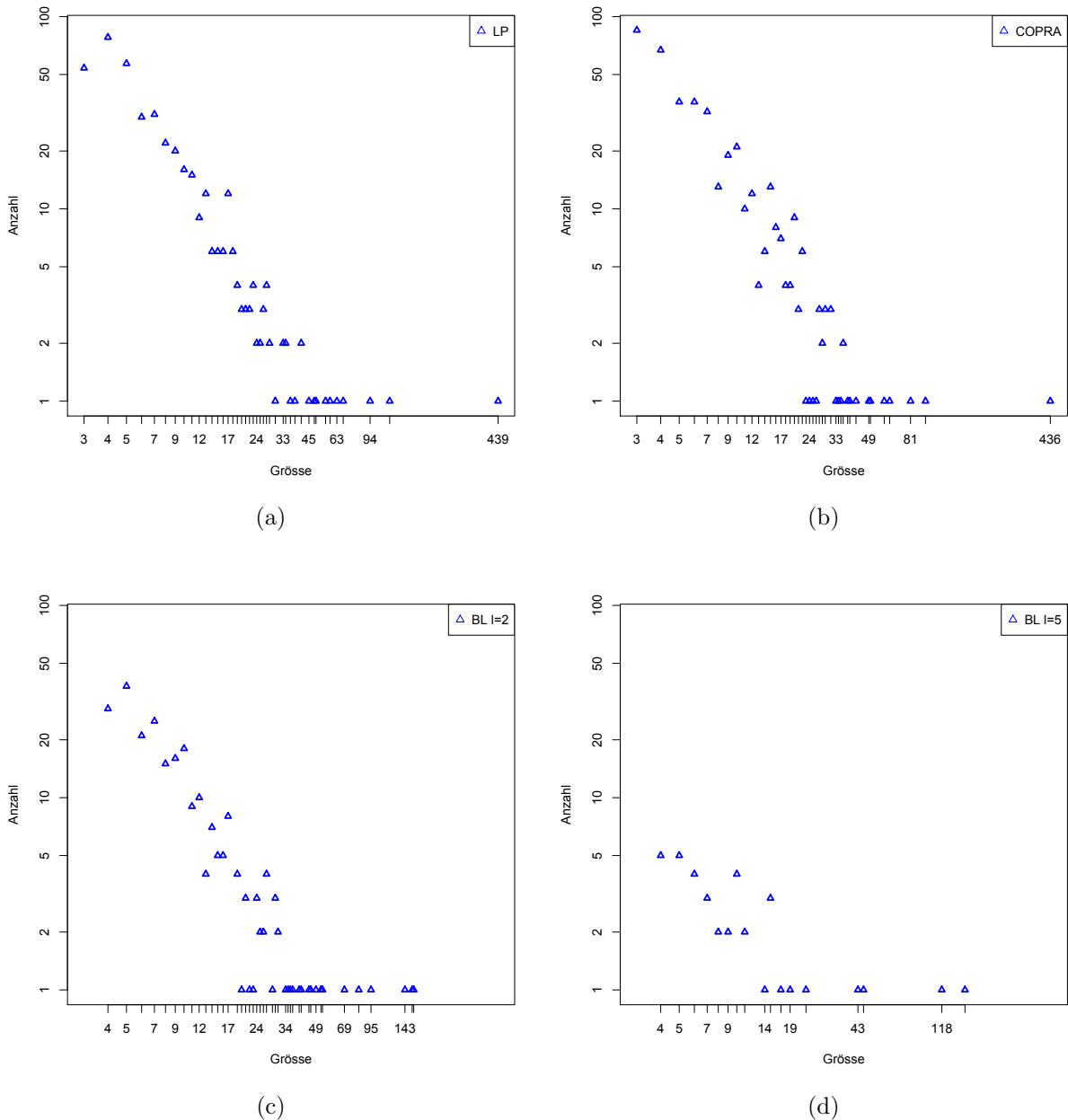


Abbildung 3.17: Verteilung der Grösse der von einer SLD dominierten Communities

Vielzahl solcher Gruppen nicht über eine gemeinsame Domain verfügen, und damit mit den hier vorliegenden Daten schlicht nicht sichtbar sind. Zwar wurde versucht, der anzunehmenden Komplexität der Gruppenzugehörigkeiten durch überlappende Communities zu begegnen. Allerdings konnte der verwendete Algorithmus keine signifikanten Überlappungseigenschaften liefern. Die Frage, inwiefern sich die einzelnen Communities sozialen Gruppen zuordnen lassen, konnte also nur unvollständig beantwortet werden.

Es zeigt sich, dass dem Web of Trust kein einzelner, einfacher Entstehungsmechanismus der hier angenommenen Form zugrundeliegt. Es kann nur unzureichend beschrieben werden als eine Ansammlung von einzelnen, einfach entstandenen Bausteinen, als die hier Communities angenommen wurden. Implizit vorausgesetzt wird

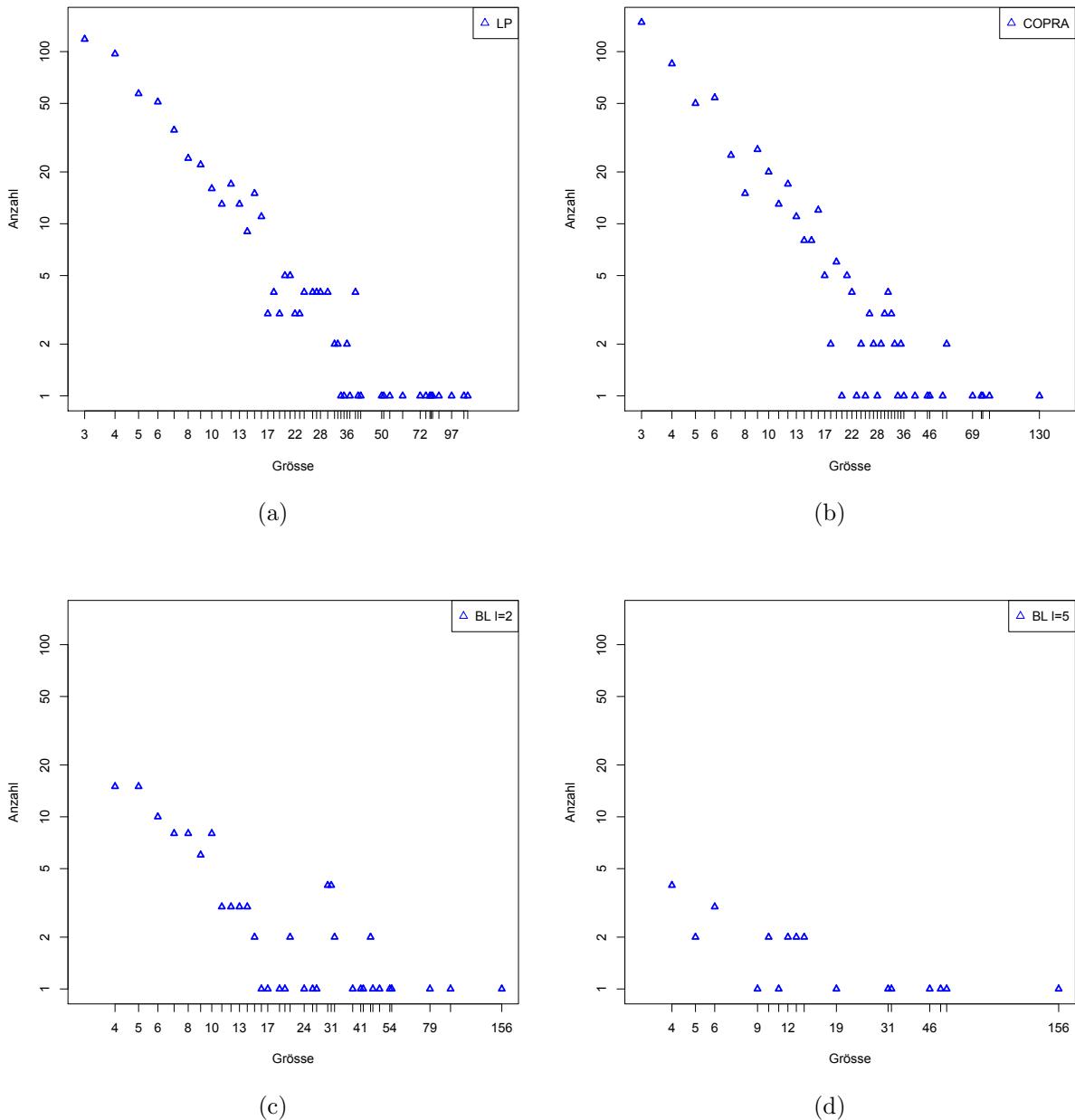


Abbildung 3.18: Verteilung der Grösse von Communities mit zeitlicher Korrelation der Signaturen

dabei auch, dass diese zum Beispiel im Fall einer Keysigning-Party nach ihrer Entstehung keine wesentliche weitere Vernetzungsaktivität zeigen. Zwar trifft dies für einen Teil der Communities – gerade die, die eine zeitliche Korrelation der Signaturen zeigen oder einer Second-Level-Domain zugeordnet werden können – durchaus zu. Für einen erheblichen Anteil der Teilnehmer – beispielsweise die Mitglieder der sehr grossen Community aus der COPRA-Berechnung – scheint die Vernetzung jedoch so vielfältig zu sein, dass solche einfachen Mechanismen nicht mehr sichtbar sind und “in der Masse verschwinden”. Bei der COPRA-Zerlegung ist es etwa naheliegend, dass die sehr grosse zentrale Community die Mehrzahl der aktiveren Teilnehmer enthält. Hier ergibt sich eine Ähnlichkeit zur Struktur der Zusammenhangskomponenten, bei

denen die grösste alle aktiven einigermassen aktiven Teilnehmer zu enthalten scheint (siehe Abschnitt 3.3.2).

Diese Punkte können anhand des Beispiels von Debian illustriert werden: Das Debian-Projekt als tatsächliche Community, in der PGP und das Web of Trust eine besonders wichtige Rolle spielen (siehe Abschnitt 1.2.4) findet sich in den Zerlegungen nicht als eigene abgrenzte Einheit wieder. Statt dessen finden sich sehr grosse Communities, in denen Debian-Entwickler einen signifikaten Anteil der Schlüssel stellen. Im Fall der COPRA-Zerlegung ist dies die bereits erwähnte Community mit ca. 21000 Mitgliedern. Von diesen verfügen 1247 über eine E-Mail-Adresse in der Domain debian.org und sind damit offensichtlich Mitglieder des Debian-Projekts. Daneben finden sich Mitglieder des Debian-Projekts in einer Vielzahl von kleineren Communities aller Grössen. Die Mitgliedschaft im Debian-Projekt scheint hier also für die Vernetzung keine so wesentliche Rolle zu spielen, dass sie die Struktur dieser real existierenden Community im Web of Trust wiederspiegelt. Selbst wenn viele Debian-Entwickler mit vielen anderen Projektmitgliedern vernetzt sind, so verfügen sie offensichtlich über genügend andere Kontakte, so dass diese Struktur nicht mehr auffindbar ist.

3.3.2 Inhaltliche Analyse der Zusammenhangskomponenten

Starke Zusammenhangskomponenten können als ein Extremfall von Communities betrachtet werden, also Mengen von Knoten, die intern stärker vernetzt sind als extern.

- strange Komponenten mit unklarer Funktion: kaos.org, ethereal.com, ...
- Bei den kleineren (<20): Viele Komponenten, die mehrere Schlüssel einer Person enthalten oder nur aus einer Person bestehen
- Komponente aus einer einzelnen Firma, keine weitere Vernetzung. Schlüssel werden vermutlich nur intern benutzt. Stellt aber einen effektiven Ersatz für eine CA zur Absicherung der internen Kommunikation dar (Serco, ponl).

Ergebniss: bei den kleineren Komponenten handelt es sich entweder um komplett geschlossene “Communities”, deren Teilnehmer von vornherein nur an der Community selbst interessiert sind oder um Gruppen, die so inaktiv sind, dass es für eine weitere Vernetzung nicht gereicht hat.

4. Zusammenfassung und Ausblick

- Datensatz: Unterschied zu anderen (Wotsap...)
- Datensatz wurde benutzt, um verschiedene Aspekte des Web of Trust zu untersuchen: Komponentenstruktur, Netzwerkstatistiken der MSCC, Community-Struktur, Verbindung mit inhaltlichen Informationen, Benutzung von Algorithmen. Wurde versucht, die Auswirkungen von Kryptoanalytischen Durchbrüchen abzuschätzen.
- viele Schluessel in Datenbank, aber nur wenige davon nehmen am Web of Trust teil.
- Grosse Mehrzahl der PGP-Benutzer (Leute mit Schluesseln, kann allerdings nicht vorausgesetzt werden, dass diese tatsaechlich benutzt werden) authentifiziert Schluessel nicht
- mesoscopic: MSCC hat ausgeprägte modulare Struktur wie viele andere soziale Netzwerke
- Communities stellen Bausteine dar, die das Gesamtnetz ergeben.
- Einige dieser Communities weisen Charakteristika auf, die eine Zuordnung zu einer sozialen Gruppe oder Keysigning-Party erlauben.
- Auflösungslimit hurts: Zusammenpacken in grosse Communities bringt zwar möglicherweise bessere Modularity, erschwert aber die inhaltliche Analyse.
- Allerdings sind diese Mechanismen nicht geeignet, um die Entstehungsdynamik insgesamt befriedigend zu erklaeren.
- Obwohl das nicht untersucht wurde, kann doch vermutet werden, dass die überwiegende Mehrheit der Schluessel in der MSCC von Personen stammt, die aus der Computersicherheit, Open-Source-Gemeinde stammen oder einen akademischen Background haben. Allerdings gibt es auch eine ganze Reihe Communities, die sich einzelnen Firmen zuordnen lassen (sony, cisco).

Schlüssel und Signaturen enthalten grundsätzlich den Zeitpunkt ihrer Erzeugung und im Keyserver-Netzwerk sind alle jemals öffentlich gemachten Schlüssel abrufbar. Im Unterschied zu vielen anderen sozialen Netzwerken ist hier die gesamte Entwicklungsgeschichte verfügbar. Der in dieser Arbeit berechnete Datensatz könnte damit ein interessanter Ansatzpunkt sein, um die Entstehungsdynamik eines komplexen sozialen Netzwerks im Detail zu untersuchen.

Literaturverzeichnis

- [BnPSDGA04] M. Boguñá, R. Pastor-Satorras, A. Díaz-Guilera und A. Arenas. Models of social networks based on social distance attachment. *Phys. Rev. E* 70(5), Nov 2004, S. 056122.
- [BrSc06] D. Brondsema und A. Schamp. Konfidi: Trust Networks Using PGP and RDF. In T. Finin, L. Kagel und D. Olmedilla (Hrsg.), *MTW*, Band 190 der *CEUR Workshop Proceedings*. CEUR-WS.org, 2006.
- [CaBH02] S. Capkun, L. Buttyán und J.-P. Hubaux. Small worlds in security systems: an analysis of the PGP certificate graph. In *NSPW '02: Proceedings of the 2002 workshop on New security paradigms*, New York, NY, USA, 2002. ACM, S. 28–35.
- [Dell07] M. Dell’Amico. Mapping Small Worlds. In M. Hauswirth, A. Wierzbicki, K. Wehrle, A. Montresor und N. Shahmehri (Hrsg.), *Peer-to-Peer Computing*. IEEE Computer Society, 2007, S. 219–228.
- [FoBa07] S. Fortunato und M. Barthelemy. Resolution limit in community detection. *Proceedings of the National Academy of Sciences* 104(1), 2007, S. 36–41.
- [Fort10] S. Fortunato. Community detection in graphs. *Physics Reports* 486(3-5), 2010, S. 75 – 174.
- [GodC09] B. H. Good, Y.-A. de Montjoye und A. Clauset. The performance of modularity maximization in practical contexts, 2009.
- [Greg10] S. Gregory. Finding overlapping communities in networks by label propagation. Arxiv preprint arXiv:0910.5516, February 2010.
- [HeGu09] J. Heikkilä und A. Gurto. Filtering SPAM in P2PSIP Communities with Web of Trust. In A. U. Schmidt und S. Lian (Hrsg.), *MobiSec*, Band 17 der *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*. Springer, 2009, S. 110–121.
- [LaFo09] A. Lancichinetti und S. Fortunato. Community detection algorithms: A comparative analysis. *Phys. Rev. E* 80(5), Nov 2009, S. 056117.
- [LeNe08] E. A. Leicht und M. E. J. Newman. Community Structure in Directed Networks. *Phys. Rev. Lett.* 100(11), Mar 2008, S. 118703.

- [McHP09] C. McDonald, P. Hawkes und J. Pieprzyk. Differential Path for SHA-1 with complexity $O(2^{52})$. Cryptology ePrint Archive: Report 2009/259 <http://eprint.iacr.org/2009/259>, June 2009.
- [MiTZ03] Y. Minsky, A. Trachtenberg und R. Zippel. Set reconciliation with nearly optimal communication complexity. *IEEE Transactions on Information Theory* 49(9), 2003, S. 2213–2218.
- [NMCM09] V. Nicosia, G. Mangioni, V. Carchiolo und M. Malgeri. Extending the definition of modularity to directed graphs with overlapping communities. *Journal of Statistical Mechanics: Theory and Experiment* 2009(03), 2009, S. P03024.
- [Noac09] A. Noack. Modularity clustering is force-directed layout. *Phys. Rev. E* 79(2), Feb 2009, S. 026102.
- [RoBe08] M. Rosvall und C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences* 105(4), 2008, S. 1118–1123.
- [SoHM05] C. Song, S. Havlin und H. A. Makse. Self-similarity of complex networks. *Nature* 433(7024), Januar 2005, S. 392–395.
- [SSAL⁺09] M. Stevens, A. Sotirov, J. Appelbaum, A. K. Lenstra, D. Molnar, D. A. Osvik und B. de Weger. Short Chosen-Prefix Collisions for MD5 and the Creation of a Rogue CA Certificate. In *CRYPTO*, 2009, S. 55–69.
- [Wiki10] Wikipedia. Pretty Good Privacy — Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/w/index.php?title=Pretty_Good_Privacy&oldid=353992410, 2010. [Online; accessed 5-April-2010].