7. Mai 2018 Studie · Web & Cloud · Digitale Identität · 23 Min. Lesezeit

Sichere Peer-to-Peer-Netze auf Basis von Selbstzertifizierung

Archiviert. Dieser Inhalt wird nicht mehr aktualisiert. Er bleibt zur Information verfügbar.

Stand 2018-05-04
Quelle https://technology2.a-sit.at/veroeffentlichungen/sichere-peer-to-peer-netze-auf-basis-von-selbstzertifizierung/

Peer-to-Peer Netze ermöglichen eine Separation zwischen Identität (Identifikator) und Ort (Netzwerkadresse). Diese Trennung ermöglicht es, virtuelle, homogene Netze auf Basis bestehender, heterogener Netzwerkstrukturen umzusetzen. Allerdings ergeben sich durch diese Separation eine Vielzahl neuer Sicherheitsrisiken, wie z.B. Vortäuschen fremder Identitäten, oder die Verbreitung falscher Routinginformationen.

Im Rahmen dieses Projekts wurden bestehende Peer-to-Peer-Konzepte um selbstzertifizierende Identifikatoren erweitert, welche einer Reihe bekannter Bedrohungen vorbeugen. Im Kern soll verhindert werden, dass nicht akkurate Informationen im Netzwerk propagiert werden. Besonderes Augenmerk wird auf die Evaluierung vollkommen dezentraler Netzwerkmodelle gelegt, welche zu keinem Zeitpunkt auf zentrale Instanzen, Zugriffskontrollen, oder andere Mechanismen setzen, welche nicht im Rahmen offener, vollständig dezentraler P2P-Netze anwendbar sind. Die Praxistauglichkeit des Ansatzes wurde mit Hilfe eines Demonstrators evaluiert.

1. Einleitung

Peer-to-Peer-Netzwerke wie beispielsweise BitTorrent [1] ermöglichen durch die Trennung von Ort und Identität die Verbindung von Geräten im Rahmen logischer Overlay-Netze auf Basis vorhandener heterogener Netzwerke. Die Adressierung von Netzwerkknoten erfolgt über logische, von der Netzwerkadresse entkoppelte Identifikatoren. Um eine Verbindung zu einem mittels Identifikator festgelegten Zielknoten aufzubauen, kommen spezielle verteilte Overlay-Routingprotokolle zu Einsatz, welche die zu einem Identifikator gehörige Netzwerkadresse auflösen. Besonders in vollständig dezentralen P2P-Netzwerken kann jedoch die Validierung von Herkunft und Authentizität von Daten (und Routinginformationen) zu einer Herausforderung werden. Seit der © A-SIT, Partner für Sicherheitsfragen ersten Konzeption solcher Systeme wurde viel Forschungsaufwand in diese Richtung betrieben [2]

[3] [4] [5] [6] [7] [8], wobei einige Lösungsansätze besonders auf Grund breiter Verfügbarkeit kostengünstiger Rechenleistung mittlerweile als problematisch angesehen werden müssen. Im Rahmen dieses Projekts wurde ein Peer-to-Peer-Netzwerkdesign auf Basis selbstzertifizierender Identifikatoren erarbeitet. Im Fokus der Arbeit stehen dabei strukturierte Peer-to-Peer Netze. Besonderes Augenmerk wurde dabei auf allgemeine Einsetzbarkeit der Lösung gelegt. Daher wurden keine Annahmen bezüglich Nutzerverhalten oder Netzwerkverkehr getroffen. Als Folge dessen musste ein möglichst allgemeines Sicherheitskonzept entwickelt werden. Ein im Rahmen dieses Projekts entwickelter Demonstrator veranschaulicht die Praxistauglichkeit des Ansatzes.

Aus diesem Entwicklungs- und Evaluierungsprozess wurde folgendes Ergebnis erzielt: Ausgehend von der Annahme, dass der (aktuell vor allem im Blockchain-Umfeld eingesetzte) Proof-of-Work-Ansatz ein probates Mittel gegen bestimmte Peer-to-Peer-spezifische Attacken ist (siehe Abschnitt 2.2), wurde zuerst eine vollständig dezentrale Lösung angestrebt. Beim Versuch, den notwendigen Skalierungsfaktor der Proof-of-Work-Maßnahme im Rahmen ausgewählter Angriffsszenarien zu errechnen, zeigte sich, dass derartige Sicherheitsmaßnahmen dem aktuellen Umfeld schlicht nicht mehr gewachsen sind. In Folge dessen ist die gesamte Klasse von Proof-of-Work-basierten Schutzmechanismen im Kontext von (strukturierten) Peer-to-Peer-Netzen neu zu bewerten. Im Zuge dieses Projekts wurde auch eine wissenschaftliche Publikation zu diesen Ergebnissen verfasst, welche im Sommer 2018 im Rahmen einer internationalen Konferenz präsentiert wird. Ein Veröffentlichungsdatum dieser Publikation ist aktuell (Stand April 2018) noch nicht bekannt. Aus diesem Grund wird die Pre-Print-Version zur Verfügung gestellt [1] .

2. Hintergrund

Ein Großteil der Konzepte und Grundlagen heute aktiv eingesetzter Peer-to-Peer-Netze stammt aus den frühen Zweitausenderjahren. Entsprechend spiegeln diese das Umfeld und die Nutzergewohnheiten aus der damaligen Zeit wider. Besonders das kollektive Bewusstsein zu ITund Netzwerksicherheit hat sich seitdem stark gewandelt. Dies liegt nicht zuletzt darin begründet, dass durch den starken Einsatz verbundener, bzw. cloudbasierter Anwendungen adäquate Sicherheitsmaßnehmen schlichtweg erforderlich sind. Eine weitere Entwicklung, die das Umfeld im Vergleich zu vor fünfzehn Jahren massiv verändert hat, ist die Popularität von leistungsstarken mobilen internetfähigen Endgeräten wie Smartphones und Tablet-Computer. Dieses Umfeld und diese Gerätelandschaft sind nicht mit allen frühen Sicherheitskonzepten vereinbar. Besonders die allgemein und vor allem extrem preiswert verfügbare Rechenleistung durch virtualisierte Recheninstanzen in der Cloud entspricht nicht mehr den Annahmen, die den Sicherheitsmodellen einiger (auch noch heute eingesetzter) Peer-to-Peer-Netzwerkarchitekturen zu Grunde liegen. Aus diesem Grund wurden bestehenden Sicherheitskonzepte für Peer-to-Peer-Netze unter Berücksichtigung aktueller Rahmenbedingungen neu evaluiert. Die Ergebnisse sind dem nachfolgenden Abschnitt zu entnehmen.

2.1. Sicherheit von Peer-to-Peer-Netzen

Generell ist festzuhalten, dass seit der Veröffentlichung der bedeutendsten Peer-to-Peer-Netzwerkdesigns wie *Kademlia* [9], *Chord* [10], oder *CAN* [11] immer wieder Bemühungen unternommen wurden, diese an sich wenig auf Sicherheit ausgerichteten Designs gegen Angriffe vieler Art zu härten. Allen (dezentralen) Peer-to-Peer Netzen ist gemein, dass das Overlay-Netzwerk betreffende Routinginformationen von den Netzwerkteilnehmern selbst erzeugt und verbreitet werden. Folglich ist die Integrität dieser Routinginformationen (ohne die das Netzwerk unmöglich Teilnehmer miteinander verbinden kann)

besonders schützenswert. Allgemein ist daher *routing table poisoning*, also das Verbreiten falscher Routinginformationen, als eine besonders kritische Angriffsklasse einzustufen. Exemplarisch für diese Kategorie von Angriffen ist die so genannte *Eclipse-Attacke* [3] zu nennen. Das Ziel dieser Attacke ist das Einkesseln, bzw. Ausgrenzen einzelner Netzwerkknoten (oder im Peer-to-Peer-Netz gespeicherter Information). Erreicht wird dies durch die gezielte Erstellung von Identifikatoren, sodass Knoten derart in der Overlay-Topologie platziert werden, dass diese ausgewählte Zielknoten vollständig einkreisen. In Folge dessen ist immer zumindest ein Knoten Teil eines jeden Routingpfads in Richtung Zielknoten. Ausgehend von dieser Position können Routinganfragen das Angriffsziel betreffend verfälscht, bzw. verworfen werden. Auf diese Art und Weise lassen sich Knoten vollständig vom restlichen Netzwerk abschneiden. Ein weiterer wichtiger Angriff im Peer-to-Peer-Kontext ist die *Sybil-Attacke* [12]. Wenn es Angreifern möglich ist, beliebig viele logische Identifikatoren (und damit beliebig viele Knoten in einem Peer-to-Peer-Netzwerk) zu generieren, können diese unerwünscht hohen Einfluss, z.B. auf das eingesetzte Routingprotokoll erlangen und somit unter anderem Eclipse-Attacken ausführen. Dieses Vortäuschen vieler unabhängiger Identitäten wird in Anlehnung an einen Fallbericht zu dissoziativer Identitätsstörung mit Titel *Sybil* als Sybil-Attacke bezeichnet. Die Verfälschung von Routinginformation kann auch dazu dienen, Man-in-the-Middle-Angriffe auszuführen, wenn Routinganfragen gezielt über von Angreifern kontrollierte Knoten umgeleitet werden. Wenn es keine Bindung von Identifikatoren an Knoten gibt, lassen sich Identifikatoren von anderen Knoten schlichtweg übernehmen, wodurch gezielt falsche Identitäten vorgetäuscht werden können. Ist die Integrität der Nachrichten, welche im Rahmen des Routingprotokolls ausgetauscht werden, generell nicht gewährleistet, lassen sich auch Netzwerkteilnehmer, die sich korrekt verhalten, dahingehend ausnutzen, Denial-of-Service-Angriffe durchzuführen. Eine spezifische Ausprägung dieser Angriffsklasse ist die so genannte *Backscatter-Attacke* [13], der ein denkbar einfaches Prinzip zu Grunde liegt: Angreifer senden eine Vielzahl von Routing-Anfragen an beliebige Knoten aus und fälschen den Absender der Nachricht. Wenn alle Nachrichten scheinbar von selben Absender stammen, wird dieser mit Antworten von ehrlichen Knoten überschwemmt. Typischerweise hat ein einzelner Knoten der Vielzahl an eingehenden Nachrichten wenig entgegenzusetzen und wird überlastet. Trotz dieser Schwächen können Peer-to-Peer-Netze robuster als zentralisierte Systeme umgesetzt werden, wenn entsprechende Gegenmaßnahmen implementiert sind. Durch die Verteilung der Funktionalität kann vor allem verhindert werden, dass es einen einzigen zentralen Angriffspunkt gibt. Des Weiteren erübrigt sich der Bedarf nach breitbandig angebundener, leistungsfähiger Hardware, um große Netzwerke versorgen zu können, da die aufzubringende Leistung von den Netzwerkteilnehmern gemeinsam zur Verfügung gestellt wird. Auch Fragen wie Vendor-Lock-In stellen sich tendenziell nicht, wenn alle Funktionalität vollständig auf die Netzwerkteilnehmerinnen und -teilnehmer verteilt ist. Um diese potentiell vorteilhafte Funktion auch tatsächlich zur Verfügung stellen zu können, hat es viele Bestrebung gegeben, Peer-to-Peer Netze um Maßnahmen gegen die zuvor beschriebenen Angriffe zu erweitern.

2.2. Sicherheitsmaßnahmen im Kontext von Peer-to-Peer-Netzen

S/Kademlia [2] ist eine solche Erweiterung des Kademlia-Designs, welche, basierend auf Selbstzertifizierung und disjunkten Routing-Pfaden, das Ziel verfolgt, Eclipse-Attacken zu verhindern. Disjunkt bedeutet in dem Fall, das Routing-Anfragen bereits mehrfachredundant abgesendet werden, und diese erst am Zielknoten konvergieren. Dabei handelt es sich um einen vollständig dezentralen Ansatz, im Rahmen dessen Teilnehmer selbst für die Erzeugung von logischen Identifikatoren zuständig sind. Um zu verhindern, dass Identifikatoren gezielt in der Nähe von bestehenden Identifikatoren generiert werden können, wird ein Proof-of-Work-Ansatz, ähnlich dem von Bitcoin, verfolgt: Ein Identifikator ergibt sich aus

einem kryptografischen Hash, welcher über den öffentlichen Teil eines public/private key pair berechnet wird. Von anderen Netzwerkteilnehmern akzeptiert werden jedoch nur Identifikatoren, welche einen Präfix bestehend aus Nullen aufweisen, der über eine (netzwerkweit statisch) definierte Mindestlänge hinausgeht. Um nachzuweisen, dass ein Knoten den behaupteten Identifikator tatsächlich generiert hat, werden alle vom ihm versendeten Nachrichten mit dem privaten Teil des zuvor erwähnten Schlüsselpaares signiert. Dies wird als Selbstzertifizierung verstanden. Im Rahmen dieses Projekts hat sich jedoch gezeigt, dass diese Gegenmaßnahme nicht mehr zeitgemäß ist, da Rechenleistung extrem billig im Rahmen von Cloud-Computing-Angeboten zur Verfügung steht. Details zu den Ergebnissen sind Abschnitt 6 zu entnehmen. Des Weiteren wurden nicht alle Vorteile von Selbstzertifizierung ausgenutzt, wodurch das Routing-Protokoll unnötig verkompliziert wurde, was zu erhöhter Netzwerklast führt. Letzteres ist besonders im Hinblick auf Denial-of-Service-Angriffe kritisch zu hinterfragen. Generell ist zu beobachten, dass sich Erweiterungen, bzw. Entwicklungen im Rahmen von Peer-to-Peer-Netzen betreffend deren Sicherheit häufig auf Sybil- und Eclipse-Attacken beziehen. Mache Eclipse-Gegenmaßnahmen setzen auch eine Resistenz gegen Sybil-Attacken voraus, um überhaupt effektiv zu arbeiten [3]. Dieser vorwiegende Fokus auf ein Problem manifestiert sich auch in wissenschaftlichen Publikationen. Levine, Shields und Margolin [6] haben mehr als neunzig Maßnahmen gegen Sybil-Attacken in fünf Kategorien eingeteilt:

- Zertifizierung auf Basis einer zentralen, vertrauenswürdigen Instanz in Sinne einer PKI
- Resource Testing (Proof-of-Work/Storage/...)
- Laufende Kosten z.B. durch kontinuierlichen, bzw. periodischen Proof-of-Work
- Trusted Computing
- Keine Gegenmaßnahme

Die Autoren kommen dabei zu dem Schluss, dass lediglich PKI-Ansätze Sybil-Attacken wirklich ausschließen können. Der Tenor, dass Proof-of-Work die Gefahr einer Sybil-Attacke zumindest hinreichend eindämmen kann, ist jedoch ebenfalls nicht haltbar; wie im Rahmen dieses Projekts festgestellt wurde (siehe Abschnitt 6). Solchen Ansätzen liegt die Annahme zu Grunde, das Rechenleistung zwischen ehrlichen Nutzern und Angreifern zumindest nicht zu krass ungleichverteilt ist. Betrachtet man jedoch die aktuellen Nutzergewohnheiten, ergibt sich bereits innerhalb der Geräte eines einzelnen Nutzers, bzw. einer einzelnen Nutzerin (Smartphone, Laptop, potentiell leistungsstarker Desktop-PC, ...) ein Ungleichgewicht. Versucht man Proof-of-Work-Ansätze auf diese Gerätelandschaft anzuwenden, lässt sich die Frage, wie viel Rechenleistung aufzubringen ist, um einen gültigen Identifikator zu erstellen, in Hinblick auf verfügbare Rechenleistung nicht ohne weiteres beantworten. Bedenkt man noch die ungleich höhere Rechenleistung, die potentielle Angreifer zur Verfügung haben, zeigt sich die Problematik dieses Konzepts noch deutlicher. Diese Situation führt dazu, dass kontinuierliche Kosten als alternative Maßnahme vollkommen ausscheiden, wenn man auch die begrenzten Akkukapazitäten von Mobilgeräten in die Bewertung möglicher Abwehrmechanismen miteinbezieht.

Eine bisher noch nicht diskutierte Klasse von Maßnahmen gegen Angriffe auf Peer-to-Peer-Netze basiert auf dem Graph der sozialen Netze von Netzwerkknoten. Auf Basis der Arbeit von Viswanath [8] zeigen Li et al. [7] jedoch, dass diese Art von Gegenmaßnahmen ebenfalls auf in der Realität nicht haltbaren Annahmen basiert. Dadurch ist auch die Effektivität von Lösungen wie *SybilGuard* [5] in Frage zu stellen. Des Weiteren kommen die Autoren ebenfalls zu dem Schluss, dass Proof-of-Work unter realitätsnahen Bedingungen nur

begrenzt wirkungsvoll sind, versuchen dem jedoch durch eine Abwandlung des Proof-of-Work Ansatzes beizukommen. Allerdings ist dieser Vorschlag auf Grund der vorliegenden Projektergebnisse ebenso in Zweifel zu ziehen. Die Basis dieser Erkenntnisse ist die nachfolgend beschriebene Peer-to-Peer-Netzwerkarchitektur.

3. Ursprüngliche Systemarchitektur

Auf Basis bekannter Angriffe auf Peer-to-Peer-Netze lässt sich eine Reihe von Bedingungen formulieren, die erfüllt sein müssen, um ebendiese Angriffe zu unterbinden:

1. Um zu verhindern, dass Identifikatoren von Angreifern übernommen werden, ist Selbstzertifizierung einzusetzen.
2. Identifikatoren sollten möglichst gleichverteilt sein, damit das gezielte Erstellen von Identifikatoren mit bestimmten Werten erschwert wird (dies erschwert Eclipse-Attacken).
3. Insgesamt ist es ausgehend von den ersten beiden Bedingungen Sinnvoll, an Identifikatoren dieselben Anforderungen wie an kryptografische Haschfunktionen zu stellen, daher sind Identifikatoren unter Zuhilfenahme kryptografischer Hashfunktionen abzuleiten.
4. Integrität und Authentizität von Nachrichten muss gewährleistet, bzw. überprüfbar sein, um die Verbreitung von falschen (Routing-)Informationen zu unterbinden.
5. Integrität und Authentizität von Absender- und Empfängerinformation in Nachrichten muss gewährleistet, bzw. überprüfbar sein, um beispielsweise DoS-Angriffe, wie Backscatter-Attacken zu verhindern.
6. Allgemein ist auf Effizienz und geringen Overhead (Bandbreite, Rechenleistungsbedarf, Speicherplatzbedarf) zu achten, um auch leistungsschwächere oder schlechter angebundene Endgeräte bedienen zu können.
7. Ausgehend von den mannigfaltigen Konsequenzen erfolgreicher Sybil-Attacken, muss ein Mindestmaß an Robustheit gegenüber dieser Angriffsklasse gewährleistet werden können.

Ausgehend von diesen Anforderungen wurde ein ursprüngliches Systemmodell abgeleitet, welches auf der Annahme beruht, dass Sybil-Attacken durch den Einsatz von Proof-of-Work-Verfahren effektiv verhindert werden können. Dieses Systemmodell ist durch die folgenden Eigenschaften definiert:

- Jeder Knoten verfügt über einen logischen, von der Netzwerkadresse unabhängigen, Identifikator
- Jeder Knoten besitzt ein selbsterstelltes public/private key pair – aus Performance-Gründen auf Basis elliptischer Kurven
- Als Identifikator kommt ein kryptografischer Hashwert zum Einsatz; dieser wird über den öffentlichen Teil des Schlüsselpaares eines Knotens berechnet.
- Ein Identifikator ist gültig, wenn er über ein Präfix aus Nullen verfügt, dessen Länge von einem netzwerkweiten Parameter definiert wird

- Alle Kommunikation ist verbindungslos und basiert auf Nachrichten (im Gegensatz zu Streams, wie z.B. im Rahmen von TCP)

o Jede Nachricht enthält einen *Message Code*, der den Typ der Nachricht signalisiert o Jede Nachricht muss vom Absender mit dessen private key signiert werden o Jede Nachricht enthält den öffentlichen Teil des public/private key pair des Absenders o Jede Nachricht enthält einen Zeitstempel o Jede Nachricht enthält die Netzwerkadresse des Absenders o Jede Nachricht enthält eine kryptografische Nonce, ähnlich einer Session-ID o Jede Nachricht enthält den Identifikator des Zielknotens

- Die Authentizität und Integrität jeder eingehenden Nachricht ist mittels Signaturprüfung zu verifizieren

- Netzwerkteilnehmer dürfen nicht eigenmächtig Informationen andere Knoten betreffend (wie z.B. die Zuordnung von logischem Identifikator zur Netzwerkadresse) erstellen, sondern lediglich zuvor vom betreffenden Knoten empfangene und signierte Nachrichten weiterleiten

- Empfänger von Nachrichten haben zu überprüfen, ob eingehende Nachrichten tatsächlich an sie adressiert sind

- Routing-Anfragen sind über disjunkte Pfade abzuarbeiten [2]

Dieser Einsatz von Selbstzertifizierung verhindert praktisch alle relevanten Angriffe auf Peer-to-Peer-Netzwerkebene, insbesondere Eclipse-Attacken, da Identifikatoren nicht gezielt generiert werden können. Allerdings ist diese Garantie wertlos, wenn Angreifer effizient beliebig viele Identifikatoren generiere können, bis ausreichend Identifikatoren vorhanden sind, die alle für einen Angriff (egal welcher Art) notwendigen Voraussetzung erfüllen. Dies steht in dierektem Zusammenhanf mit Sybil-Attacken. Daher muss der Aufwand für Sybil-Attacken durch die entsprechende Skalierung des Identifikator-Präfix' so hoch werden, dass solche Angriffe unrentabel werden. Eine derartige Skalierung ist zwar prinzipiell möglich, allerdings wird dadurch auch der notwendige Aufwand für die Erzeugung gültiger Identifikatoren für ehrliche Netzwerkteilnehmer unverhältnismäßig hoch, was zu mangelnder Akzeptanz, bzw. völliger Unbenutzbarkeit führt. Die Details hierzu werden im nachfolgenden Abschnitt erläutert. Zuvor wird jedoch skizziert, wie das vorgestellte System (ausgehend von der vielzitierten Annahme, dass Proof-of-Work ohne Auswirkung auf ehrliche Benutzer entsprechend skaliert werden kann) Angriffe verhindert.

Als Routing-Protokoll kommt eine Weiterentwicklung von S/Kademlia zum Einsatz, das seinerseits eine Erweiterung von Kademlia darstellt. Die Besonderheit des hier vorgestellten Systems besteht vor allem in der Struktur der Routingtabellen, deren Aufgabe es ist, die logischen Identifikatoren von Knoten mit den zugehörigen Netzwerkadressen zu verknüpfen. Da diese Information, bzw. ihre Weitergabe die Grundlage des Routing-Protokolls ist, ist auch ihre Integrität und Authentizität besonders kritisch. Entgegen bestehender Systeme werden in Routing-Tabellen nicht nur Identifikator und IP-Adresse gespeichert, sondern eingehende Nachrichten im Ganzen, welche bei der Beantwortung von Routing-Anfragen unverändert weitergegeben werden. Grund dafür ist, dass alle Nachrichten signiert sind, und Signaturen dadurch auch nach Weitergabe an andere Knoten prüfbar bleiben und somit die Authentizität und Integrität aller Informationen zu jedem Zeitpunkt verifiziert werden kann. Alle notwendigen Informationen, um Identifikatoren der Netzwerkadresse des zugehörigen Knotens zuzuordnen, sind bereits in Nachrichten enthalten. Durch die Bindung von Identifikatoren an den Besitz privater Schlüssel können Identifikatoren nicht von Angreifern übernommen werden; der Einsatz von Signaturen auf Basis dieses Schlüssels setzt

dessen Besitz voraus. Dadurch, dass Identifikatoren vom zugehörigen öffentlichen Schlüssel abgeleitet sind und dieser in jeder Nachricht enthalten sein muss, ist eine Signaturprüfung möglich, im Rahmen derer gleichzeitig der Absender einer Nachricht zweifelsfrei verifiziert werden kann. In Kombination mit erforderlicher Angabe des Empfängers von Nachrichten können auch Backscatter-Attacken verhindert werden. Setzt man voraus, dass alle Nachrichten auch die Netzwerkadresse des Absenders enthalten (wie unter anderem vom Kademlia-Design vorgesehen), verbreiten sich dadurch nur garantiert akkurate Routinginformationen im Netzwerk. Zusammenfassend kann jeder Knoten nur für sich selbst sprechen, da die Routinginformationen betreffend einen Knoten nur von ebendiesem Knoten als Teil von Nachrichten generiert werden können (zumindest, wenn diese vom restlichen Netzwerk auch akzeptiert werden sollen). Der Einsatz von Zeitstempeln regelt einerseits zu jedem Zeitpunkt eindeutig, welche Information die aktuellste ist. Da jeder Knoten nur für sich selbst sprechen kann, ist jedoch keine gemeinsame Netzwerkzeit notwendig. In Folge dessen wird auch das Verarbeiten obsoleter Nachrichten weitestgehend verhindert. Durch den Einsatz kryptografischer Hashfunktionen wird eine Gleichverteilung von Identifikatoren garantiert. Des Weiteren sind Identifikatoren dadurch nicht vorhersehbar, was Eclipse-Attacken erschwert, bzw. verhindert, sobald Sybil-Attacken nicht mehr durchführbar sind. Versucht man letzteres auf Basis von Proof-of-Work (ein Präfix bestehend aus Nullen als Teil des Identifikators) umzusetzen, stellt sich die Frage nach der Skalierung.

Der tatsächlich notwendige Aufwand, um Angriffen unterschiedlichen Ausmaßes beizukommen, lässt sich auf Basis der Netzwerkgröße berechnen. Dieser Vorgang wird im nachfolgenden Abschnitt beschrieben.

4. Proof-of-Work als Schutz vor Eclipse- und Sybil-Attacken

Die Effektivität, bzw. die entsprechende Skalierung von Proof-of-Work im Rahmen der Identifikatorgenerierung lässt sich an Hand der in Tabelle 1 angeführten Parameter berechnen.

k ≡ systemweiter Replikationsfaktor; gibt die Anzahl disjunkter Routing-Pfade an b ≡ Identifikatorlänge in Bit $S = 2^{[b]}$ ≡ Größe des Identifikatorraums c ≡ Kosten, um einen Identifikator zu generieren N ≡ tatsächliche Netzwerkgröße, bzw. Anzahl von Knoten $d = SN$ ≡ durchschnittliche kürzeste Distanz zwischen zwei Knoten *Tabelle 1: Netzwerkparameter* Zielt man nun darauf ab, einen einzelnen Knoten durch eine Eclipse-Attacke vom Rest des Netzwerks so abzuschotten, dass dieser nicht mehr erreichbar ist, müssen *ausreichend viele* bösartige Knoten nahe am Zielknoten angesiedelt werden. Nahe genug bedeutet in diesem Fall, dass die Identifikatoren der für den Angriff notwendigen Knoten näher am Identifikator des Zielknotens liegen, als der nächste (ehrliche) Nachbar. Diese allgemeinen Zusammenhänge werden nachfolgend hergeleitet. Anschließend werden deren Auswirkungen diskutiert.

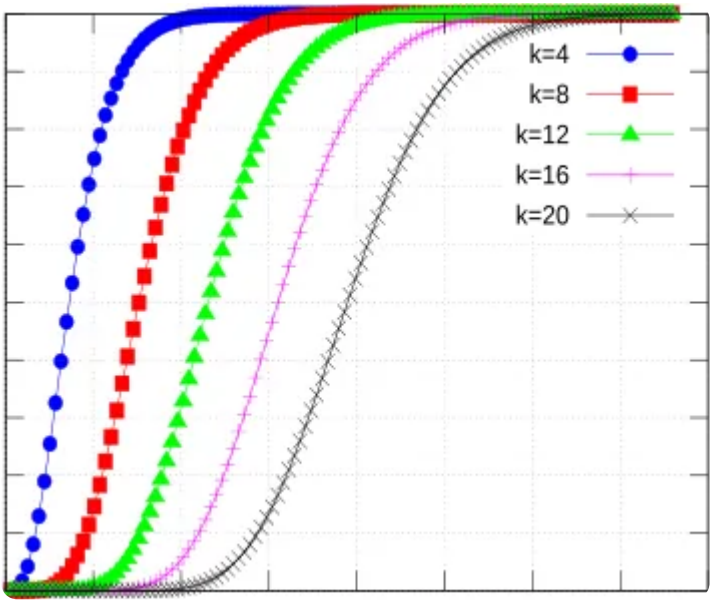
4.1. Zusammenhänge zwischen Netzwerkparametern und Angriffsaufwand

Unter der Annahme, dass der Identifikatorraum mehrere Größenordnungen über der tatsächlichen Netzwerkgröße liegt, lässt sich die Wahrscheinlichkeit, dass ein generierter Identifikator (dessen Wert nicht vorhersehbar, bzw. vorberechenbar ist) die notwendige Distanz unterschreitet, an Hand von Formel 1 berechnen:

$$\sum_{i=1}^{d-1} p(\text{dist} = i) = p(\text{dist} < d) = p(\text{Nachbar}) \quad [1]$$

[1] S [=] N N ≈ [S] [S] N [× 1] S S *Formel 1: Wahrscheinlichkeit, einen Identifikator in Unmittelbarer Nähe eines Zielknotens zu generieren* Berücksichtigt man des Weiteren, dass Routing-Anfragen in über k viele disjunkte Pfade abgearbeitet werden, müssen zumindest k viele Knoten in unmittelbarer Nähe des Zielknotens generiert werden [2]. Dabei handelt es sich um einen stochastischen Prozess. Auf den ersten Blick entspricht dieses wiederholte Generieren von Identifikatoren innerhalb der Gesamtmenge aller möglichen Identifikatoren (in der Hoffnung, dass die Erstellten ein Wunschkriterium erfüllen) einem *Ziehen mit Zurücklegen* . Mit Zurücklegen, deshalb, weil sich nach jedem Ziehen (Generieren) die Gesamtmenge aller möglichen Ausgänge für den nachfolgenden Versuch nicht verändert. Das wiederholte Generieren bis zu einer bestimmten Anzahl von Erfolgen (Identifikatoren in der Nähre des Zielknotens) lässt sich durch einen Bernoulli-Prozess auf Basis einer Binomialverteilung annähern. Allerdings muss noch für die Möglichkeit kompensiert werden, dass ein relevanter Identifikator mehrmals generiert wird (was einem *Ziehen ohne Zurücklegen* entspricht). Insgesamt lässt sich damit die Erfolgswahrscheinlichkeit für eine Eclipse-Attacke (also für die Erstellung k vieler Identifikatoren näher am Zielknoten als der nächste Nachbar) nach n Versuchen als *Ziehen ohne Zurücklegen* und einem Kompensationsfaktor an Hand von Formel 2 berechnen.

k−1 (1 − [1] k k $p(Eclipse)$ = (1 −∑(([n] N [)] i n−i S [)] (1 −)) i N [)] N [)] ‿ i=0 N [)]) × (∏ { [S] {) N S [)] ‿ i=1 i [) (1] N [S] N [−i+ 1) (N] S Wahrscheinlichkeit ≥k unterschiedliche Identifikatoren zu generieren Kompensationsfaktor (Identifikatoren unterschiedlich) *Formel 2: Wahrscheinlichkeit für ausreichend Identifikatoren für eine Eclipse-Attacke nach n Versuchen* Dieser Zusammenhang gilt für das vorgestellte Peer-to-Peer-Netzwerk, kann jedoch auf Grund der tatsächlichen Gleichverteilung generierter Identifikatoren und der Tatsache, dass der in Formel 2 definierte Prozess nicht abgekürzt, bzw. die Erfolgswahrscheinlichkeit nicht erhöht werden kann, als allgemeine Messlatte herangezogen werden. Die grundsätzliche Widerstandsfähigkeit gegen Eclipse-Attacken jedes Peer-to-Peer-Netzwerks, welches dem S/Kademlia-Modell folgt und disjunkte Pfade für Routinganfragen, sowie Selbstzertifizierung vorschreibt, lässt sich auf diese Art modellieren. Stellt man dies den Kosten für die Erzeugung eines Identifikators gegenüber, lassen sich die Gesamtkosten für eine erfolgreiche Eclipse-Attacke ableiten. Als Anhaltspunkt für einen Identifikatorraum der Größe 2 [256] dient Abbildung 1. Hier zeigt sich eine nahezu lineare Abhängigkeit zwischen Netzwerkgröße und der notwendigen Versuche, um ausreichend Nachbarn zu generieren. Diese Abhängigkeit wird noch deutlicher erkennbar, wenn man die Erfolgswahrscheinlichkeit in Abhängigkeit von für k und der Netzwerkgröße visualisiert (siehe Abbildung 2).



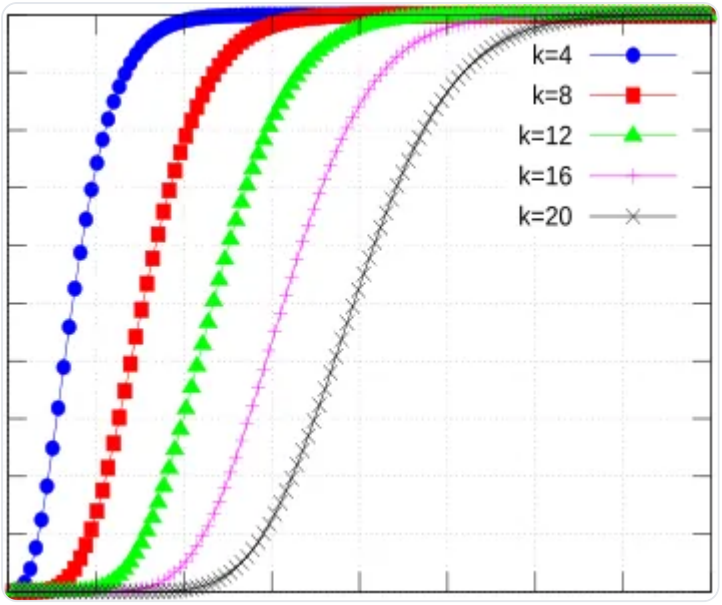
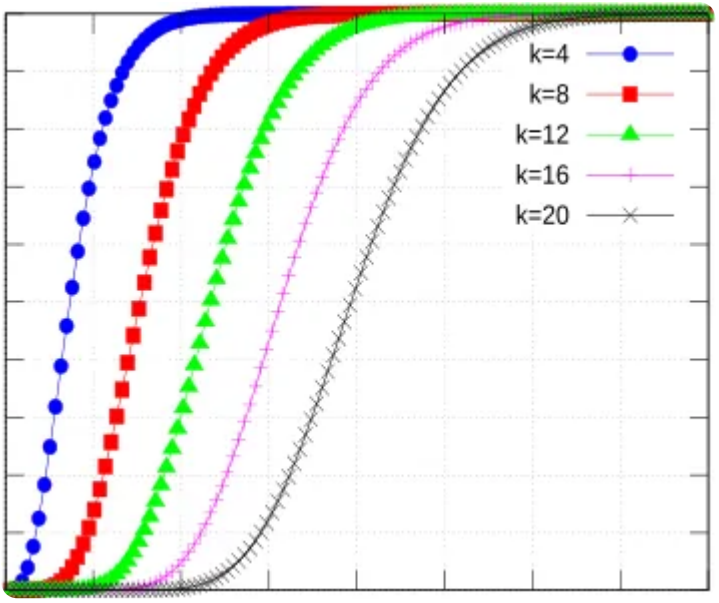
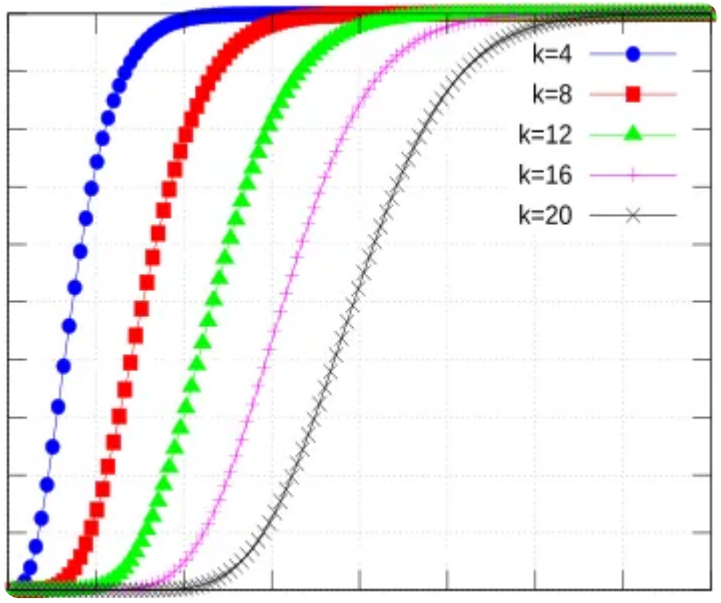
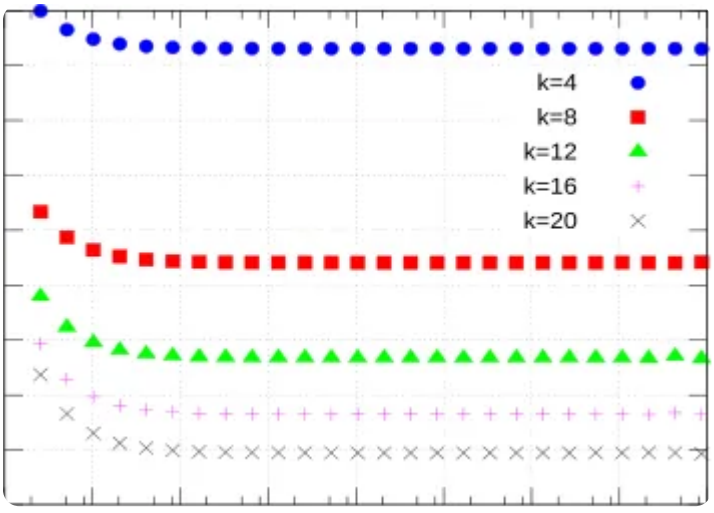
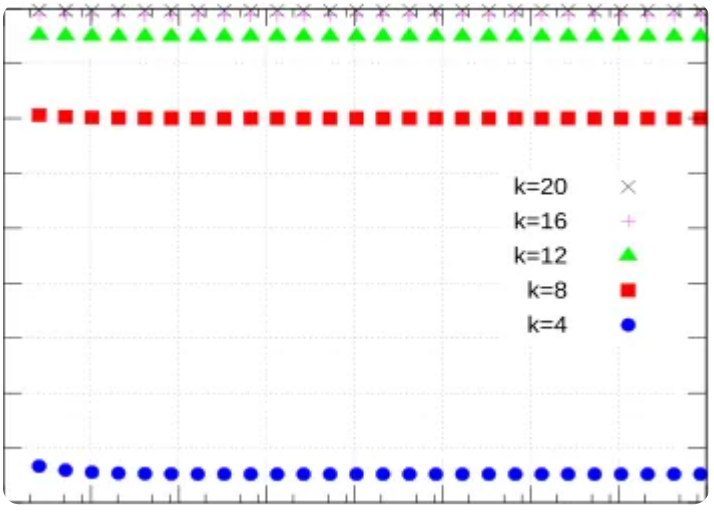


Abbildung 1: Erfolgswahrscheinlichkeit für k viele Nachbarn in für unterschiedliche Netzwerkgrößen Abbildung 2: Wahrscheinlichkeit für k viele Nachbarn bei $k\times N$ und $2k\times N$ Versuchen nach Netzwerkgrößen Die in Abbildung 2 dargestellten Verhältnisse zwischen Netzwerkparametern und dem Aufwand, die notwendigen Identifikatoren zu generieren, liefern die folgende Erkenntnis: Es reicht aus $k\times N$ viele Identifikatoren zu generieren, um bereits mit fünfzigprozentiger Wahrscheinlichkeit die für eine Eclipse-Attacke notwendigen Identifikatoren generiert zu haben. Generiert man die doppelte Menge Identifikatoren, finden sich darunter nahezu sicher ausreichend viele Identifikatoren, um den Zeitknoten vollständig einzukreisen und in weiterer Folge abkapseln zu können.









Wie zuvor erwähnt, muss noch die Frage nach den Kosten eines solchen Angriffs beantwortet werden, um die tatsächlichen Auswirkungen der eben dargelegten Zusammenhänge abschätzen zu können. Typischerweise werden für die Abschätzung von Angriffskosten die Kosten von Cloud-Computing-

Angebote herangezogen. Stellt sich heraus, dass die erwarteten Kosten für einen Angriff zu niedrig sind, kann die Identifikatorerzeugung mittels Proof-of-Work-Skalierungsfaktor angepasst werden. Details hierzu sind dem nachfolgenden Abschnitt zu entnehmen.

4.2. (Un-)Tauglichkeit von Proof-of-Work als Schutzmaßnahme

Über den Proof-of-Work-Skalierungsfaktor der in Abschnitt 3 vorgestellten Systemarchitektur, können die Kosten für die Erstellung von Identifikatoren beliebig erhöht werden. Allerding muss in diesem Zusammenhang die Auswirkung auf ehrliche Netzwerkteilnehmer berücksichtigt werden. Auf Basis eines Cloud-Computing-Angebots, dessen Leistung zwischen der aktueller Computer und aktueller Smartphones anzusiedeln ist [2], wurde errechnet, welche Angriffskosten in Abhängigkeit der Kosten für die Erzeugung eines einzigen Identifikators entstehen. Gleichzeitig eignet sich die ausgewählte Cloud-Computing-Instanz auf Grund ihrer Leistung ebenfalls um die Auswirkung unterschiedlicher Skalierungsfaktoren auf durchschnittliche Nutzerinnen und Nutzer abzuschätzen. Einerseits sind die Ergebnisse dieser Gegenüberstellung unabhängig von der Implementierung und damit auf alle vergleichbaren Peer-to-Peer-Netze (wie z.B. S/Kademlia und *IPFS [3]*) anwendbar. Andererseits erlaubt die Berechnung auf Basis einer vorhandenen Implementierung eine Reproduzierung der konkreten Ergebnisse. Aus diesem Grund wurde das in Tabelle 2 dargestellte Szenario auf Basis des im Rahmen dieses Projekts entwickelten Demonstrators herangezogen.

k 20 : systemweiter Replikationsfaktor b 256 : Identifikatorlänge in Bit S= 2 [b] 2 [256] : Größe des Identifikatorraums K h 0,004€ : Stündliche Kosten der Cloud-Instanz IDs/s ≈12.000 : Generierbare Identifikatoren pro Sekunde Sig/s ≈9.900 : Signierbare Nachrichten pro Sekunde Ver/s ≈3.200 : Verifizierbare Nachrichten pro Sekunde c 10 [–10] € : Kosten, um einen Identifikator zu generieren N 1.000.000 : tatsächliche Netzwerkgröße, bzw. Anzahl von Knoten *Tabelle 2: Parametrisierung des Evaluierungsszenarios auf Basis der Demonstratorimplementierung* Ausgehend von Abbildung 2 folgt, dass 2 × 20 × 10 [6] = 4 × 10 [7] Identifikatoren generiert werden müssen, um nahezu sicher k= 20 Identifikatoren in direkter Nachbarschaft eines zuvor ausgewählten Zielknotens zu erhalten. Die Kosten hierfür sind vernachlässigbar, daher muss der Skalierungsfaktor so angepasst werden, dass derartige Angriffe das Budget von Angreifern sprengen. Dabei stellt sich jedoch heraus, dass die Erzeugung eines einzigen Identifikators (je nach Angreifer-Budget) in Folge dessen unverhältnismäßig lange dauert (siehe Tabelle 3). Somit ist davon auszugehen, dass eine auf diese Art garantierte Robustheit gegenüber Angriffen nicht in einem alltagstauglichen System, welches auch auf breite Akzeptanz stößt, resultiert.

Tabelle 3: Unterschiedliche Skalierungsfaktoren und Angriffskosten Diese Beobachtung stellt Proof-of-Work-basierte Identifikatorgenerierung, wie sie in zahllosen wissenschaftlichen Publikationen als probates Mittel gegen Sybil- und Eclipse-Attacken genannt wird (siehe [6]), in Frage. Dies gilt auch für Sybil-Attacken im engeren Sinne: Ausgehend von zehn 2 https://hetzner.cloud/ 3 https://ipfs.io/

Dauer Identifikatorerzeugung	1 Min	5 Min	15 Min
Länge Identifikatorpräfix	20 Bit	22 Bit	23 Bit
Skalierungsfaktor	1.048.576	4.194.304	8.388.608
$K_{Angriff}$	€ 4.194,30	€ 16.777,22	€ 33.554,43

Nachrichten pro Knoten und Sekunde, belaufen sich die Kosten, um eine Million Knoten zu betreiben, auf Basis von Tabelle 2 auf ca. 33 €/h. Eine naheliegende Erklärung für die Untauglichkeit einer vielfach propagierten Maßnahme findet sich in der Preisentwicklung von Cloud-Computing-Ressourcen. In den vergangenen Jahren kam es (trotz teilweiser Marktkonsolidierung) zu einem regelrechten Preissturz. In Folge dessen ist es sehr kostengünstig möglich, ein krasses Ungleichgewicht zwischen der Rechenleistung, die ein einzelnes Endgerät, bzw. Nutzerinnen und Nutzer für den Betrieb eines einzelnen Knotens zur Verfügung haben, und der für Angreifer leistbaren Rechenleistung, herzustellen. Dadurch ist eine fundamentale Annahme, nämlich eine nicht zu krasse Ungleichverteilung von Ressourcen zwischen einzelnen Nutzern und potentiellen Angreifen, eindeutig verletzt, wodurch Proof-of-Work-basierte Sicherheitsmechanismen versagen. Als einzige anwendungsunabhängige Lösung für dieses Problem bleibt laut Levine, Shields und Margolin lediglich der Einsatz von Zertifizierung im Sinne einer PKI. Entsprechend wurde eine neue Systemarchitektur abgeleitet, welche im nachfolgenden Abschnitt beschrieben ist.

5. Angepasste Systemarchitektur

Die grundsätzliche Systemarchitektur, sowie das verwendete Routing-Protokoll werden von der Unwirksamkeit von Proof-of-Work als Maßnahme gegen Eclipse- und Sybil-Attacken nicht in Frage gestellt. Unverändert kommen selbstzertifizierende Identifikatoren und signierte Nachrichten zum Einsatz. Statt Proof-of-Work wird jedoch eine zentrale vertrauenswürdige Instanz angenommen, welche die Identifikatoren jedes Knotens signiert. Eine entscheidende Bedingung an diesen Ansatz ist jedoch, dass diese Signaturen nicht automatisiert beziehbar sein dürfen. Eine Möglichkeit, dies sicher zu stellen, ist beispielswiese der Einsatz von CAPTCHAs, wie z.B. *reCAPTCHA [4]* . Zusammenfassend ergeben sich folgende Differenzen zur ursprünglich definierten Systemarchitektur:

- Jeder Knoten verfügt über ein Zertifikat einer vertrauenswürdigen Instanz
- Identifikatoren müssen von dieser vertrauenswürdigen Instanz mittels zum Zertifikat passender digitaler Signatur zertifiziert werden
- Jede Nachricht enthält diese Signatur, sowie eine Referenz auf das zur Signaturprüfung notwendige Zertifikat; hierbei handelt es sich somit um eine klassische PKI, wie sie auch im Rahmen von TLS zu Einsatz kommt
- Jeder Knoten hat die Gültigkeit der Signatur zu überprüfen
- Es gibt keine Anforderungen an den Wert des Identifikators im Sinne eines Proof-of-Work

Durch diese Anpassungen werden Eclipse- und Sybil-Attacken auf Kosten vollständiger Dezentralisierung effektiv verhindert. Zusammenfassend sind Angriffe auf die im Rahmen dieses Projekts entwickelten Peer-to-Peer-Netzwerkarchitektur damit nicht mehr rentabel. Selbst ein weiterer Preisverfall von Rechenleistung hat keinen Einfluss auf die Sicherheit des Systems. Weiters wird auch die Netzwerklast nicht unnötig erhöht, da keine zusätzlichen Nachrichten ausgetauscht werden müssen, um die Authentizität und Integrität von im Netzwerk verteilten Informationen zu garantieren. Abgesehen von diesen funktionalen Anforderungen, wurde auch der Ressourcenbedarf des entwickelten Demonstrators ermittelt, welcher durch den Einsatz von Signaturen, bzw. deren Verifizierung beim Senden, bzw. Empfangen von Nachrichten entsteht. Tabelle 4 verdeutlicht, dass dieser Ansatz auf aktuellen Endgeräten durchaus alltagstauglich ist.

Anzahl Operationen ID Generierung Signieren Signaturverifikation 100.000 8.385 ms 10.268 ms 31.648 ms **500.000** 41.245 ms 50.609 ms 157.823 ms **1.000.000** 83.399 ms 101.600 ms 317.111 ms **Durchschnitt/s** ≈12.000 OPs ≈9.900 OPs ≈3.200 OPs

Tabelle 4: Ressourcenbedarf einzelner Operationen

6. Fazit

Ausgehend von der Annahme, dass Proof-of-Work-basierte Erzeugung von selbstzertifizierenden Identifikatoren im Rahmen strukturierter Peer-to-Peer-Netze hinreichenden Schutz gegen typische Peer-to-Peer-spezifische Angriffe bietet, wurde versucht, diese Sicherheitsmarge, bzw. die Kosten für erfolgreiche Angriffe zu quantifizieren. Im Zuge dessen wurde ein Demonstrator entwickelt, welcher ein entsprechendes Peer-to-Peer-Netz auf Basis von Kademlia, bzw. einer Weiterentwicklung von *S/Kademlia* umsetzt. Tatsächlich hat sich jedoch gezeigt, dass die aktuell extrem niedrigen Preise für Cloud-Computing-Ressourcen Angreifern einen Vorteil verschaffen, welcher nicht sinnvoll kompensiert werden kann. Vor diesem Hintergrund hat sich herausgestellt, dass Proof-of-Work-basierte Identifikatorgenerierung, wie sie vielfach propagiert wird, nicht mehr zeitgemäß ist und andere Methoden gewählt werden müssen, um Peer-to-Peer-Netzwerke ausreichen sicher zu gestalten.

4 https://www.google.com/recaptcha/intro/android.html

Referenzen

[1] B. Cohen, „The BitTorrent Protocol Specification," 11 10 2013. [Online]. Available:

http://www.bittorrent.org/beps/bep_0003.html. [Zugriff am 24 04 2018].

[2] I. Baumgart und S. Mies, „S/Kademlia: A practicable approach towards secure key-based routing," in *2007 International Conference on Parallel and Distributed Systems*, 2007.

[3] Atul Singh et al., „Defending Against Eclipse Attacks on Overlay Networks," in *Proceedings of the 11th Workshop on ACM SIGOPS European Workshop*, Leuven, ACM, 2004.

[4] R. Fantacci et al., „Avoiding Eclipse Attacks on Kad/Kademlia: An Identity Based Approach,"

in *2009 IEEE International Conference on Communications*, IEEE, 2009.

[5] Haifeng Yu et al., „SybilGuard: Defending Against Sybil Attacks via Social Networks," in *Proceedings of the 2006 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Pisa, ACM, 2006, pp. 267-278.

[6] B. N. Levine, C. Shields und N. B. Margolin, „A Survey of Solutions to the Sybil Attack,"

Amherst, 2006.

[7] Frank Li et al., „SybilControl: Practical Sybil Defense with Computational Puzzles," in *Proceedings of the Seventh ACM Workshop on Scalable Trusted Computing*, Raleigh, ACM, 2012, pp. 67-78.

[8] Bimal Viswanath et al., „An Analysis of Social Network-based Sybil Defenses," in *Proceedings of the ACM SIGCOMM 2010 Conference*, Neu Delhi, ACM, 2010, pp. 363-374.

[9] P. Maymounkov und D. Mazières, „Kademlia: A Peer-to-Peer Information System Based on the XOR Metric," in *Peer-to-Peer Systems*, Springer, 2002, pp. 53-65.

[10] Ion Stoica et al., „Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications," in *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, San Diego, ACM, 2001, pp. 149-160.

[11] Sylvia Ratnasamy et al, „A Scalable Content-addressable Network," in *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, San Diego, ACM, 2001, pp. 161-172.

[12] J. R. Douceur, „The Sybil Attack," in *Peer-to-Peer Systems*, Springer, 2002, pp. 251-260.

[13] David Moore et al., „Inferring Internet Denial-of-service Activity," in *ACM ransactions on Computer Systems 24.2*, ACM, 2006, pp. 115-139.