Zentrum für sichere Informationstechnologie – Austria
Secure Information Technology Center – Austria

A-SIT

Seidlgasse 22 / 9, 1030 Wien
Tel.: (+43 1) 503 19 63–0
Fax: (+43 1) 503 19 63–66

Inffeldgasse 16a, 8010 Graz
Tel.: (+43 316) 873-5514
Fax: (+43 316) 873-5520

http://www.a-sit.at
E-Mail: office@a-sit.at

DVR: 1035461
ZVR: 948166612
UID: ATU60778947

SICHERHEITSANALYSE AKTUELLER PEER-TO-PEER-NETZE

Version 1.0 vom 21.05.2019
Alexander Marsalek – alexander.marsalek@a-sit.at
Bernd Prünster – bernd.pruenster@a-sit.at

Abstract/Zusammenfassung: Dieses Projekt analysiert die Sicherheitskonzepte aktuell im Einsatz befindlicher Peer-to-Peer-Netze. Unter anderem durch den Kryptowährungshype der letzten Jahre erfahren Dezentralisierungsgedanken vermehrt Zuspruch, wobei Sicherheitskonzepte im Rahmen moderner Peer-to-Peer-Systeme im Vergleich zu frühen Entwicklungen einen integralen Bestandteil der Systemarchitektur bilden. Auf Grund von aktuellen Erkenntnissen zum Thema Peer-to-Peer-Sicherheit, welche aus einem vorangegangen A-SIT-Projekt hervorgingen, ergaben sich jedoch Zweifel an der Effektivität insbesondere von Proof-of-Work-basierter Identifikatorgenerierung im Rahmen dezentraler Peer-to-Peer-Netze. Im Speziellen wurde IPFS, das als vielseitig einsetzbarer Peer-to-Peer-Netzwerklayer und Motor für Dezentralisierungsbestrebungen positioniert wird, auf potentielle Schwachstellen analysiert. Dabei konnten Teilerfolge in Richtung praktisch relevanter Angriffe erzielt werden und teilweise auch Netzwerkknoten vom Rest des Netzwerks abgeschnitten werden.
Darüber hinaus gibt dieses Dokument einen Überblick über die Sicherheitskonzepte von BitTorrent und Bitcoin. Abschließend wird ein neues Sicherheitskonzept für dezentrale Peer-to-Peer-Netze auf Basis von Android-Remote-Attestation-Verfahren vorgestellt, dessen technische Details im Rahmen einer aus diesem Projekt hervorgegangenen wissenschaftlichen Publikation ausgearbeitet wurden, welche im Rahmen der 16th International Conference on Security and Cryptography 2019 präsentiert wird.

Inhaltsverzeichnis

1. Einleitung

Im Zentrum dieses Projekts steht die Sicherheitsanalyse aktuell im Einsatz befindlicher Peer-to-Peer-Netze. Dies umfasst sowohl die Netzwerklayer moderner, dezentraler Kryptowährungen, als auch Filesharing-Netzwerke wie *BitTorrent* [1]. Eine Sonderstellung nimmt *IPFS*, das *Interplanetary File System* [2], ein. Im Gegensatz zu vielen Anwendungen, welche notgedrungen einen Peer-to-Peer-Netzwerklayer benötigen, ist das Ziel von IPFS eine möglichst anwendungsneutrale Basis für verteilte/dezentrale Anwendungen zu schaffen. IPFS wird auch zunehmend entsprechend genutzt, wodurch eventuelle Schwachstellen im System folglich Auswirkungen auf alle Anwendungen hätten, welche IPFS als Netzwerklayer einsetzen. Aus diesem Grund wurde IPFS die meiste Aufmerksamkeit gewidmet.
Die Sicherheitseigenschaften von Peer-to-Peer-Netzwerken, welche als Unterbau für Kryptowährungen eingesetzt werden, wurden exemplarisch an Hand von *Bitcoin* [3] analysiert. Da Filesharing über BitTorrent wieder im Aufwind begriffen ist [4], wurde auch dieses Peer-to-Peer-Netzwerk behandelt.

Im nachfolgenden Abschnitt werden Peer-to-Peer-Netze an Hand ihrer grundlegenden Eigenschaften kategorisiert. Anschließend wird in Abschnitt 3 ein Überblick über peer-to-peer-spezifische Angriffsszenarien und Gegenmaßnahmen gegeben. Da diese Thematik bereits in vorangegangen A-SIT-Projekten ausführlich behandelt wurde, wird im Detail auf die entsprechenden Projektberichte verwiesen, anstatt alle Einzelheiten erneut aufzuführen. Ausgehend von diesen Grundlagen geht Abschnitt 4.1 auf das Sicherheitskonzept von BitTorrent ein. In Abschnitt 4.2 wird IPFS im Detail auf potentielle Schwachstellen analysiert und mögliche Angriffe werden diskutiert. Den Abschluss der Analysen bildet Abschnitt 4.3 mit einem Überblick über die Sicherheitskonzepte von Peer-to-Peer-Netzen im Kryptowährungskontext. Anschließend wird in Abschnitt 5 ein neues Sicherheitskonzept auf Basis von Remote-Attestation-Verfahren vorgestellt. Die Ergebnisse dieses Projekts werden in Abschnitt 6 zusammengefasst.

2. Kategorisierung von Peer-to-Peer Netzwerkkonzepten

Peer-to-Peer-Netzwerkkonzepte können in zwei zueinander orthogonalen Dimensionen unterschieden werden. Grundsätzlich gibt es strukturell gegensätzliche Konzepte Peer-to-Peer-Kommunikation umzusetzen: Strukturierte Peer-to-Peer-Netze erlauben nur wohldefinierte Operationen im Rahmen von Netzwerkmanagement und bieten im Gegenzug hohe Effizienz. Unstrukturierte Peer-to-Peer-Netze hingegen beschränken sich nicht nur auf die Bereitstellung eines Overlays, sondern ermöglichen die Umsetzung beliebig komplexer Operationen direkt auf Netzwerkebene. Details hierzu werden im nachfolgenden Abschnitt behandelt. Unabhängig vom strukturellen Aufbau eines Peer-to-Peer-Netzwerks kann die Organisation beliebig zentral oder dezentral erfolgen. Dieser Aspekt wird in Abschnitt 2.2 behandelt.

2.1. Strukturell unterschiedliche Peer-to-Peer-Netzwerkkonzepte

Unabhängig von ihren konkreten Eigenschaften verfolgen Peer-to-Peer-Netze das vordergründige Ziel, Teilnehmerinnen und Teilnehmer auf Basis eines (typischerweise homogenen) logischen Overlays miteinander zu vernetzen. Die IP-Netzwerktopologie rückt dabei in den Hintergrund. Stattdessen wird eine Abstraktionsschicht eingeführt, die lediglich logische Identifikationen zur

Identifikation und Lokalisierung von Netzwerkknoten verwendet, welche transparent auf IP-Adressen abbilden und so die Komplexität der tatsächlichen Netzwerktopologie versteckt. Dies kann entweder nach einem strikten Schema im Rahmen strukturierter Peer-to-Peer-Netze, oder organisch, ohne globale Struktur im Rahmen unstrukturierter Peer-to-Peer-Netze erfolgen.

2.1.1. Strukturierte Peer-to-Peer-Netze

Strukturierte Peer-to-Peer-Netze bilden oftmals die Basis komplexerer verteilter Anwendungen und dienen primär als Abstraktionsschicht der IP-Netzwerktopologie. Vom Netzwerk selbst werden typischerweise lediglich Operationen wie das Lokalisieren von Netzwerkknoten oder Daten (und Pointern zu Daten) unterstützt. Durch die Beschränkung auf wohldefinierte Operationen lässt sich effizientes Routing mit definierten Umlaufzeiten umsetzen. Die verteilten Routingprotokolle von strukturierten Peer-to-Peer-Designs wie *Chord* [5], *CAN* [6] und *Kademlia* [7] garantieren beispielsweise, dass nie mehr als $\mathcal{O}(\log n)$ viele Hops benötigt werden, um einen Knoten zu lokalisieren. Gleichzeitig wird die Netzwerklast minimiert, da Routinganfragen immer auf dem kürzesten Weg zum Ziel konvergieren. Möglich wird dies durch den Einsatz wohldefinierter Distanzfunktionen, die auf die Identifikatoren von Knoten und Daten angewandt werden. Geht eine Routinganfrage an einem Knoten ein, geht aus dessen Routingtabelle direkt hervor, welche ihm bekannten Knoten näher am Ziel sind. Entsprechend können optimale Pfade zur Lokalisierung von Knoten angewandt werden. Darüber hinaus lässt sich an Hand solch wohldefinierter Distanzfunktionen und der garantierten Konvergenz von Routinganfragen auch zweifelsfrei feststellen, wenn ein gesuchter Knoten aktuell nicht Teil des Netzwerks ist.
Höhere Funktionen, wie z.B. Volltextsuche auf der im Netzwerk gespeicherten Datenbasis, können auf diese Weise nicht modelliert werden und benötigen entsprechend zusätzliche Applikationslogik, welche über den Funktionsumfang eines strukturierten Peer-to-Peer-Netzwerks hinausgeht.

2.1.2. Unstrukturierte Peer-to-Peer-Netze

Unstrukturierte Peer-to-Peer-Netze erzwingen im Gegensatz zu ihren strukturierten Gegenstücken keine wohldefinierte Netzwerkstruktur und garantieren weder Maximalumlaufzeiten, Effizienz, noch Konvergenz von (Routing-)Anfragen. Gleichzeitig gibt es jedoch konzeptionell de-facto keine Limitierungen, was die von einem unstrukturierten Design zur Verfügung gestellte Funktionalität angeht. Anfragen zur Lokalisierung von Knoten können ebenso leicht umgesetzt werden, wie Volltextsuchen auf dem gesamten im Netzwerk vorhandenen Datenbestand. Knoten selbst können auch nach beliebigen Kriterien lokalisiert werden, da keine Notwendigkeit besteht, dass Anfragen einem bestimmten Schema folgen. Vielmehr werden eingehende Anfragen – egal wie komplex – bestmöglich beantwortet, ohne dass klar ist, an welche anderen Knoten diese Anfrage weitergeleitet werden muss, um beantwortet werden zu können.
Im einfachsten Fall wird dies mittels Flooding gelöst, wodurch sowohl direkt, als auch durch unstrukturierte Mehrfachantworten hohe Netzwerklast entstehen kann. Optimierungen, wie z.B. *Random-Walks* [8], dämmen dieses Problem ein, ändern jedoch nichts am Grundprinzip. Tatsächlich ist von vornherein nicht klar, ob eine Anfrage jemals erfüllt wird, wie viele Knoten beteiligt sein werden, oder wie lange die Beantwortung benötigen wird.
Nichtsdestotrotz haben unstrukturierte Peer-to-Peer-Netze eine Daseinsberechtigung. Ihre Umsetzung ist konzeptionell einfach, komplexere Funktionalität kann direkt auf Peer-to-Peer-Netzwerkebene umgesetzt werden und unterscheidet sich prinzipiell nicht von einfachen Routinganfragen. Darüber hinaus müssen auf Grund des ohnehin weniger deterministischen Verhaltens a-priori weniger Annahmen über das Verhalten anderer Netzwerkteilnehmer und Netzwerkteilnehmerinnen getroffen werden. Typischerweise kommen unstrukturierte Peer-to-Peer-Netze im Rahmen von Kryptowährungen zum Einsatz, da deren Sicherheits- und Integritätsgarantien auf höherer Ebene garantiert werden und ohnehin hauptsächlich Broadcasting zum Einsatz kommt.

2.2. Unterscheidung basierend auf Zentralisierungsgrad

Zusätzlich zur Netzwerkstruktur, können Peer-to-Peer-Netze an Hand ihres (De-)zentralisierungsgrads klassifiziert werden. Hierbei reicht das Spektrum von vollkommen dezentral organisierten

Netzen, welche ohne zentrale Instanz auskommen, bis hin zu zentralisierten Designs, welche konzeptionell eher in der Nähe traditioneller Client-Server-Architekturen anzusiedeln sind, als im engeren Peer-to-Peer-Umfeld.

2.2.1. Zentralisierte Peer-to-Peer-Netze

Im Rahmen zentralisierter Peer-to-Peer-Netzwerke findet zwar der Datenaustausch direkt zwischen Netzwerkknoten (und damit dezentral) statt, Netzwerkbeitritte und die Verarbeitung von Anfragen innerhalb des Netzwerks werden jedoch von einer oder einigen wenigen zentralen Instanzen durchgeführt.
Als Beispiel für derartige Netzwerke sind Echtzeit-Videokommunikationsdienste zu nennen. Im Rahmen solcher Anwendungen werden im Sinne geringer Latenz und der Entlastung zentraler Server direkte Verbindungen zwischen kommunizierenden Parteien aufgebaut, über die Bild und Ton übertragen werden. Funktionalität wie Account-Verwaltung und Orchestrierung von Kommunikationskanälen zwischen Teilnehmerinnen und Teilnehmern wird jedoch von einer zentralen Instanz zur Verfügung gestellt.
Zwar ergibt sich durch dieses Design auf Basis einer zentralen Instanz einerseits ein Single Point of Failure, andererseits können Zugangskontrollen einfach umgesetzt werden und dieselben Sicherheitskonzepte wie im Rahmen traditioneller Client-Server-Anwendungen angewandt werden, um Angriffe abzuwenden und Missbrauch zu verhindern. Auch der Bootstrapping-Prozess, also der initiale Verbindungsaufbau, bzw. der initiale Aufbau des Netzwerks, kann konzeptionell einfach umgesetzt werden, indem jeder Knoten bei Verbindungsaufbau die zentrale Instanz kontaktiert.
Auf Grund dieser Eigenschaften wurden derartige System im Rahmen dieses Projekts nicht behandelt, da in diesem Zusammenhang kaum Peer-to-Peer-spezifische Sicherheitsaspekte zum Tragen kommen.

2.2.2. Dezentrale Peer-to-Peer-Netze

Dezentrale Peer-to-Peer-Netze zeichnen sich durch die vollkommene Abwesenheit zentraler (Kontroll-)Instanzen aus. Funktional handelt es sich dadurch um ein vollkommen homogenes Netzwerk, dessen Knoten alle gleichberechtigt sind und gemeinsam die Netzwerkfunktionalität zur Verfügung stellen. Um Verbindungen zwischen Knoten herzustellen, werden Anfragen nach der IP-Adresse von Zielknoten an bekannte Netzwerkknoten versendet, welche diese ihrerseits weiterleiten, bis der Knoten mit dem entsprechenden Identifikator (bzw. dessen IP-Adresse) lokalisiert werden kann. Im Rahmen dieses Prozesses sind alle Knoten gleichberechtigt und gleich wichtig, ohne dass Abhängigkeiten zu zentralen Instanzen benötigt werden. Durch diese offene Organisation lässt sich beispielsweise Zensur nur schwierig durchsetzen, was oftmals als Vorteil dezentraler Peer-to-Peer-Netze genannt wird. Der Bootstrapping-Prozess ist ohne die Notwendigkeit, eine zentrale Instanz kontaktieren zu müssen, ebenso dynamisch umsetzbar, sodass es zwar vorab bekannte Einstiegspunkte für den initialen Verbindungsaufbau geben muss, prinzipiell aber jeder beliebige, einmal bekannte Knoten als Kontakt- und Einstiegspunkt verwendet werden kann. Gleichzeitig stellt sich angesichts der Abwesenheit einer zentralen Instanz die Frage, wie Fehlverhalten detektiert, verhindert und/oder sanktioniert werden soll.

2.2.3. Hybride Peer-to-Peer-Netze

Hybride Peer-to-Peer-Netze stellen einen Kompromiss bezüglich des Dezentralisierungsgrads dar und weisen einzelnen Knoten (sogenannten Super-Nodes oder Super-Peers) mehr Verantwortung im Netzwerk zu. Dabei ist es zweitrangig, wie diese Auswahl getroffen wird, da dies stark anwendungsabhängig ist. Die Organisation solcher Netze ist potentiell dezentral umgesetzt, d.h. es gibt nicht notwendigerweise zentrale Instanzen, welche den Zugang zum Netzwerk regeln. Eine Hauptaufgabe von Super-Nodes ist, das Netzwerk zu stützen. Dies kann zum Beispiel im unstrukturierten Fall bedeuten, dass Super-Nodes, eine tragende Rolle im Routing-Prozess einnehmen und Anfragen bevorzugt untereinander austauschen um reguläre Knoten zu entlasten. Diese hybride Architektur ist insbesondere unter Berücksichtigung der typischerweise heterogenen Ressourcenverteilung unter den Teilnehmerinnen und Teilnehmern eines Peer-to-Peer-Netzes interessant. Im Gegensatz zu vollständig dezentral organisierten Netzwerken ergeben sich durch

den erhöhten Einfluss von Super-Nodes auch zusätzliche Möglichkeiten, Angriffe auf das Netzwerk zu detektieren.
Details zu peer-to-peer-spezifischen Sicherheitsaspekten, Angriffen und Gegenmaßnahmen werden im Nachfolgenden Abschnitt zusammengefasst. Der Fokus liegt dabei auf dezentralen Peer-to-Peer-Netzen.

3. Peer-to-peer-spezifische Angriffe und Gegenmaßnahmen

Besonders in dezentral organisierten Peer-to-Peer-Netzen können Angriffe – wenn überhaupt – nur bekämpft, aber kaum verhindert werden. Die Thematik an sich wurde bereits in vorangegangenen A-SIT-Projekten (u.A. [9]) behandelt, weshalb dieser Abschnitt lediglich zwei signifikante Attacken und Gegenmaßnahmen kurz umreißt.

3.1. Sybil-Attacke

Die Sybil-Attacke beschreibt das Auftreten einer Partei unter vielen, scheinbar unabhängigen Identitäten innerhalb eines Peer-to-Peer-Netzwerks, was dazu führen kann, dass diese einen unverhältnismäßig hohen Einfluss z.B. auf das verteilte Routingprotokoll erlangt und so disruptive Aktionen ausführen kann [10]. Dies ist unter anderem problematisch, da sicherheitskritische Aspekte im Peer-to-Peer-Umfeld oftmals auf einer angenommenen Unabhängigkeit der Teilnehmerinnen und Teilnehmern voneinander beruhen. Wird diese Annahme verletzt, halten darauf aufbauende Sicherheitsgarantien nicht mehr. Beispielsweise werden Routinganfragen an mehrere Netzwerkknoten gestellt, damit einzelne bösartige Teilnehmerinnen und Teilnehmer die verteilten Routingprotokolle in Peer-to-Peer-Netzen nicht destabilisieren. Im Fall einer Sybil-Attacke wird dies jedoch auch für einzelne Angreiferinnen und Angreifer potentiell möglich, bzw. können Sybil-Attacken, wie nachfolgend beschrieben, die Basis für weitere Angriffe bilden.

3.2. Eclipse-Attacke

Eine Eclipse-Attacke beschreibt das umzingeln, bzw. Ausgrenzen einzelner Knoten eines Peer-to-Peer-Netzwerks [11]. Im Fall strukturierter Peer-to-Peer-Netze mit wohldefinierter Distanzfunktion zwischen Netzwerkknoten kann dies beispielsweise erreicht werden, sofern Identifikatoren frei gewählt, oder effizient generiert werden können. Ist ein Opfer ausgewählt, werden vom Angreifer, bzw. der Angreiferin, auf Peer-to-Peer-Netzwerkebene gezielt Knoten um das Opfer platziert, sodass Anfragen immer diese Knoten passieren. Erheblich erleichtert wird diese Art des Angriffs, wenn Sybil-Attacken möglich sind, da somit keine Kooperation von ansonsten unabhängigen Angreiferinnen und Angreifen nötig ist, sondern eine, bzw. einige wenige böswillige Parteien dem Netzwerk gegenüber als viele, scheinbar voneinander unabhängige Knoten auftreten können.

3.3. Gegenmaßnahmen

Im Fall beider Attacken schafft eine Beschränkung über die Anzahl der Identifikatoren, bzw. Netzwerkknoten, die von einer einzelnen Partei betrieben werden können, Abhilfe. Dies wurde bereits früh erkannt und Zentralisierung, sowie Resource-Testing (wie z.B. Proof-of-Work-basierte Identifikatorgenerierung) wurden als mögliche Gegenmaßnahmen vorgeschlagen [12]. In jedem Fall kommt eine Form von Identifikatorzertifizierung zum Einsatz. D.h. der Besitz eines Identifikators wird mittels digitaler Signaturverfahren so nachgewiesen, dass dies von jedem Teilnehmer und jeder Teilnehmerin im Netzwerk überprüft werden kann. Im Fall eines zentralisierten Peer-to-Peer-Netzwerks kann dies auch über eine PKI (in Form zertifikatsbasierter Identifikatoren und Prozeduren ähnlich wie bei der Ausstellung von TLS-Zertifikaten) erfolgen. Im vollständig dezentralen Fall gab es bisher keine zufriedenstellenden Lösungen für dieses Problem, was bereits in einem vorangegangen A-SIT-Projekt zum Thema [9] behandelt wurde. Im Rahmen dieses Projekts wurde ein Lösungsvorschlag auf Basis von Trusted-Computing-Ansätzen im Android-Umfeld erarbeitet, welcher in Abschnitt 5 vorgestellt wird. Zuvor werden jedoch bestehende, bereits im Einsatz befindliche Systeme analysiert.

4. Analyse aktueller Peer-to-Peer-Netzwerke

Aktuell erfahren Peer-to-Peer-Netze und Dezentralisierungsgedanken wieder vermehrt Zulauf, was auch am Kryptowährungshype liegt. Exemplarisch werden in diesem Abschnitt drei konkrete Peer-to-Peer-Systeme auf die Sicherheit ihres Peer-to-Peer-Netzwerklayers (und nicht vornehmlich auf die zur Verfügung gestellte Funktionalität auf höheren Abstraktionsschichten) hin analysiert.

4.1. BitTorrent

BitTorrent [1] ist ein Peer-to-Peer-Filesharing-System, dessen Sicherheitskonzept entsprechend stark auf das effiziente Verteilen von Dateien, bzw. Dateifragmenten ausgerichtet ist. Diesem Ziel folgend sollen vor allem zwei Arten von unehrlichem, bzw. unfairem Verhalten verhindert werden:

1. So genanntes *Leeching*, das reine Herunterladen von Daten, ohne im Gegenzug dem restlichen Netzwerk bereits heruntergeladene Daten zur Verfügung zu stellen

2. Die Verteilung kompromittierter Daten

Ersteres wird versucht zu verhindern, indem Clients Daten anderen Teilnehmerinnen und Teilnehmern vornehmlich zur Verfügung stellen, wenn diese selbst bereits heruntergeladene Daten ihrerseits zur Verfügung stellen. Tatsächlich ist jeder Client auf die Maximierung der Downloadgeschwindigkeit aus, und entscheidet autonom, an wen welche Teile bereits heruntergeladener Daten weitergegeben werden. Dabei werden Daten auch Teilnehmerinnen und Teilnehmern zur Verfügung gestellt, mit denen zuvor nicht interagiert wurde, in der Hoffnung, dass dies zu einem späteren Zeitpunkt von der Gegenseite vergütet wird [13]. Dadurch soll ein Pareto-Optimum im Netzwerk bezüglich der Datenverteilung erreicht werden.
In der Praxis funktioniert diese Strategie nur bedingt, und es ist (zumindest bei gut verfügbaren Daten, welche von vielen Teilnehmerinnen und Teilnehmern zur Verfügung gestellt werden) durchaus möglich, Leeching zu betreiben [14]. Tatsächlich scheint das BitTorrent-Netzwerk zu großen Teilen vom Altruismus einiger gut angebundener, freigiebiger Teilnehmerinnen und Teilnehmer getragen zu werden und funktioniert weitgehend reibungslos, da viele Teilnehmerinnen und Teilnehmer schlicht nicht versuchen, sich eigennützig zu verhalten [15].

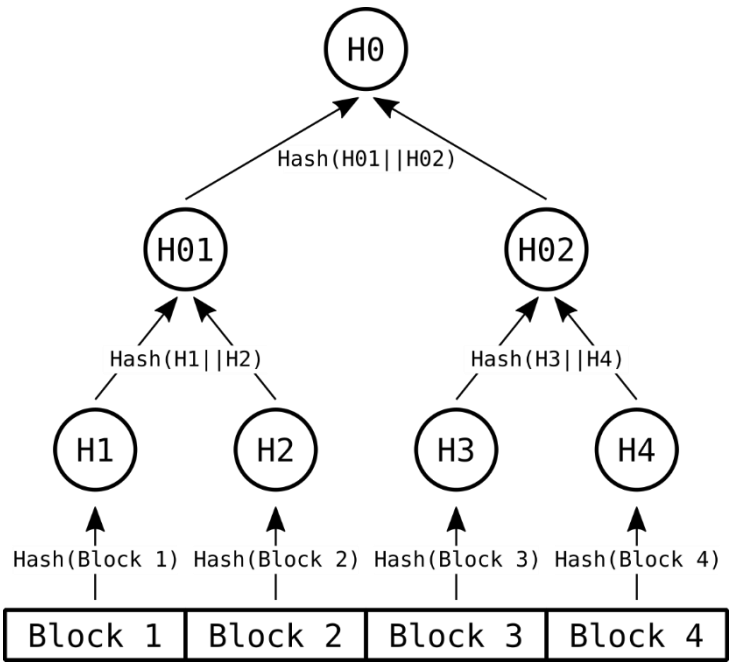

Abbildung 1: Struktur eines Merkle-Baums

Die Verteilung kompromittierter Daten wird technisch durch Datenrepräsentation als Merkle-Baum verhindert: Daten werden in Segmente eingeteilt, auf die eine kryptografische Hashfunktion angewandt wird. Rekursiv werden anschließend (im Fall eines Binärbaums) immer zwei Hashwerte erneut gehashed, bis eine Baumstruktur entsteht (siehe Abbildung 1). Verteilt werden Daten im Netzwerk, indem die Baumwurzel referenziert wird. Die Integrität aller Segmente kann somit auf Basis der Baumwurzel überprüft werden, womit sich kompromittierte Daten nicht verbreiten. Problematischer ist im BitTorrent-Kontext vielmehr das Einschleusen von (bewusst) trügerischen Daten (welche technisch korrekt als Merkle-Baum abgebildet wurden), die als legitim angepriesen werden.

Die Sicherheitsanforderungen an die eigentliche Peer-to-Peer-Netzwerkschicht sind im Rahmen von BitTorrent gering. Grund dafür ist unter anderem, dass es abseits der Verteilung von Daten keine Funktionalität gibt, welche dies erfordern würde. Leistungen wie Suche und Ankündigung von Daten werden an Webseiten und Foren ausgelagert und müssen nicht vom Netzwerk erbracht werden. Eine gemeinsame Datenbasis über alle Teilnehmerinnen und Teilnehmer gibt es nicht.
Nichtsdestotrotz bietet BitTorrent auch die Funktionalität einer verteilten Hashtabelle (*Distributed Hash Table, DHT*) nach dem Kademlia-Konzept, allerdings nahezu ohne nennenswerte Sicherheitsmaßnahmen. Identifikatoren sind schlicht Zufallswerte und können folglich vollkommen frei gewählt werden. Es gibt daher keine Möglichkeit zu verhindern, dass sich mehrere Teilnehmerinnen und Teilnehmer (freiwillig, oder durch „Identitätsdiebstahl“) ein und denselben Identifikator teilen. Auch wurde beobachtet, dass laufend Angriffe auf Basis dieser Designentscheidung auf das Netzwerk ausgeführt werden [8]. Direkte Beeinträchtigungen auf Filesharing haben jedoch hauptsächlich Social-Engineering-Ansätze, im Rahmen derer trügerische Daten eingeschleust werden, deren wahrer Inhalt sich erst zeitversetzt zeigt. Dies ist insbesondere bei interaktiven Inhalten, wie z.B. Computerspielen oder Software effektiv: Primär lässt sich in diesem Fall der Zeitraum vom Verteilen der Daten bis zum Bekanntwerden des Täuschungsmanövers länger ausdehnen als dies beispielsweise im Fall von kurzen Textdateien möglich ist. Infolgedessen wird eine manipulierte Version im Netzwerk populär, da die erhaltenen und verteilten Daten vorerst integer scheinen. Alternative (korrekte) Versionen der Daten hingegen verlieren im Netzwerk an Popularität und werden dadurch verdrängt – potentiell bevor die Verfälschung der eigentlich gefragten Daten auffällt.

4.2. IPFS

IPFS, das Interplanetary File System [2], wird als vielfältig einsetzbarer open-source Peer-to-Peer-Layer positioniert. Dabei handelt es sich um ein dezentrales, strukturiertes Peer-to-Peer-Netzwerk auf *S/Kademlia*-Basis [12]. Entsprechend findet das Sicherheitsmodell von S/Kademlia auch im Rahmen von IPFS Anwendung. Vereinfacht formuliert kann das Netzwerk als praktisch sicher angesehen werden, so lange keine Sybil- und Eclipse-Attacken durchgeführt werden können. Konkret darf es nicht möglich sein, dass ein einzelner Angreifer oder eine einzelne Angreiferin beliebig viele Identifikatoren generiert und/oder Netzwerkknoten betreibt. Da S/Kademlia auf selbstzertifizierenden Identifikatoren aufbaut, können Identifikatoren nicht frei gewählt werden, sondern sind vom öffentlichen Teil eines Public-Private-Key-Pair abgeleitet. Der private Schlüssel wird benötigt um ausgehende Nachrichten zu signieren, wodurch der „Besitz“ eines Identifikators kryptografisch nachgewiesen werden kann. Laut dem offiziellen Whitepaper des Projekts wird die Generierung großer Identifikatormengen durch einen Proof-of-Work-Ansatz verhindert.
Darüber hinaus bedient sich Kademlia für die Verteilung von Daten BitTorrent-Konzepten (siehe Abschnitt 4.1), was verhindern soll, dass sich Teilnehmerinnen und Teilnehmer (zu) eigennützig verhalten.

Die IPFS-Implementierung selbst ist plattformunabhängig in der Programmiersprache Go umgesetzt und bietet darüber hinaus eine HTTP-API an, welche von beliebigen Programmiersprachen aus genutzt werden kann. Innerhalb des Netzwerks werden langlebige TCP-Verbindungen zur Kommunikation genutzt. Die Funktionalität von IPFS geht über die eines reinen Overlays hinaus und bietet unter anderem Operationen auf einer globalen, verteilten Hashtabelle an, stellt ein eigenes Nameservice zur Verfügung und einen kollaborativ verwalteten gerichteten, azyklischen Merkle-Graph (*Merkle-DAG*) zur verteilten Datenspeicherung bereit. Dabei handelt es sich um ein Konstrukt ähnlich einem Merkle-Baum, allerdings mit weniger restriktiver Graphstruktur. mit IPFS wird auch

als sichere Peer-to-Peer-Netzwerkimplementierung positioniert, wobei an mehreren Stellen auf diversen Projektseiten explizit darauf hingewiesen wird, dass sich das Projekt noch in Entwicklung befindet, und daher Vorsicht geboten ist, wenn IPFS-basierte Anwendungen im Produktivbetrieb eingesetzt werden sollen.
Presseartikel [16] über IPFS und im Rahmen dieses Projekts durchgeführte Crawl-Versuche lassen nichtsdestotrotz darauf schließen, dass das IPFS-Netzwerk aktuell (Stand Q1 2019) mehrere Millionen Knoten umfasst. Ausgehend von einem vorangegangen A-SIT-Projekt [9], einer daraus resultierenden wissenschaftlichen Publikation zum Thema sichere Peer-to-Peer-Netze [17] und der Tatsache, dass IPFS laut Whitepaper auf einem Proof-of-Work-Ansatz gegen Sybil- und Eclipse-Attacken setzt [2], wurde die Effektivität dieser Maßnahmen im Rahmen dieses Projekts auf die Probe gestellt.

4.2.1. Identifikation potentieller Schwachstellen

Höhere IPFS-Funktionen bauen auf einem funktionierenden Overlay und dessen Sicherheitsgarantien auf. Entsprechend weitreichend wären die Konsequenzen von Schwachstellen auf unterster Ebene des IPFS-Stacks; gleichzeitig handelt es sich dabei um eine konzeptionell einfache Komponente des Gesamtsystems, welche keine Analysen komplexerer Subsysteme erfordern. Aus diesem Grund wurde der Analysefokus auf diese elementaren Eigenschaften gelegt.

Als Grundlage für potentielle Angriffe wurde im ersten Schritt die Generierung von Identifikatoren analysiert. Anders als im Whitepaper behauptet, können Identifikatoren praktisch beliebig generiert werden, da keinerlei Resource-Testing (wie z.B. Proof-of-Work) zum Einsatz kommt. Erschwerend kommt hinzu, dass IPFS standardmäßig RSA-basierte selbstzertifizierende Identifikatoren einsetzt, jedoch Identifikatoren auf Basis elliptischer Kurven unterstützt. Da Elliptic-Curve-Kryptografie auf Grund kürzerer Schlüssellängen weit weniger rechenintensiv umgesetzt werden kann, führt dies zu einer Situation, in der Angreifer de-facto effizienter neue Identifikatoren generieren können als Durchschnittsnutzer und –nutzerinnen. Beim Crawlen des IPFS-Adressraums hat sich diese Vermutung insofern bestätigt, als dass festgestellt werden konnte, dass ECC-basierte Identifikatoren keinen signifikanten Anteil am überwiegend von RSA-basierten Identifikatoren besetzten Identifikatorraum haben.

4.2.2. Angriffsszenarien

Ausgangspunkt für alle im Rahmen dieses Projekts durchgeführten Angriffsversuche ist die Identifikatorgenerierung, welche im nachfolgenden Abschnitt detailliert beschrieben wird. Auf Basis der zuvor skizzierten effizienten Generierung von Identifikatoren ergeben sich unter anderem folgende Angriffsszenarien:

- Sybil-Attacke: Ausgehend von einer Vielzahl (wahllos) generierter Identifikatoren wird eine große Anzahl von Knoten betrieben, welche dem Netzwerk gegenüber als unabhängige Instanzen auftreten.
- Eclipse-Attacke: Im Rahmen (oder nach) der Identifikatorgenerierung werden Identifikatoren nach ihrer Distanz zu einem Zielknoten sortiert, sodass gezielt nur so viele Knoten wie notwendig in der unmittelbaren Nachbarschaft des Zielknotens betreiben werden, um diesen einzukesseln.
- Zensur: Analog zur Eclipse-Attacke werden nicht Knoten, sondern Daten durch das gezielte Betreiben von Knoten in deren Nachbarschaft eingekesselt, um so den Zugriff darauf zu verhindern.

Umgesetzt wurde die dafür notwendige Identifikatorgenerierung auf Basis des IPFS-Sourcecodes, bzw. mittels im Rahmen von IPFS eingesetzter Module, um Kompatibilität sicherzustellen. Insbesondere konnte so eine korrekte Identifikatorrepräsentation und der Einsatz einer unveränderten Distanzfunktion sichergestellt werden.

4.2.3. Gezielte Identifikatorgenerierung

IPFS ist in hohem Maß modular aufgebaut, wodurch sich gezielt einzelne Komponenten für die Generierung von Identifikatoren verwenden lassen, ohne dass unnötiger Overhead durch den Betrieb vollwertiger IPFS-Instanzen entsteht. Konkret wurden Identifikatoren auf Ed25519-Basis erzeugt, da die zu Grunde liegende Implementierung hocheffizient arbeitet und so auch auf durchschnittlich leistungsfähigen Computern Millionen von Identifikatoren binnen weniger Tage erstellt werden können. Der Zeitaufwand für die bloße Erstellung von (zufälligen) Identifikatoren ist somit im Angriffskontext vernachlässigbar.
Um gezielt Identifikatoren in der Nähe eines Knotens zu generieren, müssen diese jedoch nach ihrer Distanz zu einem zuvor definierten Zielknoten sortiert werden. Dabei wurde die Distanz zwischen jedem generierten (zufälligen) Identifikator zum Zielknoten unmittelbar nach dessen Erstellung überprüft, um auch immer die Gesamtmenge der zu sortierenden Identifikatoren zu jedem Zeitpunkt minimal zu halten. Darüber hinaus wurde auf *Goroutinen* (ein integrales Feature der Programmiersprache Go zur effizienten Parallelisierung) zurückgegriffen. Einmalig generierte Identifikatoren können auf Grund der offenen, dezentralen Struktur von IPFS beliebig (wieder)verwendet werden.

4.2.4. Proof-of-Concept Sybil-Attacke

IPFS baut in der Standardkonfiguration bereits ohne Benutzerinteraktion hunderte Netzwerkverbindungen auf, wodurch Angriffe, welche den Betrieb mehrerer Instanzen voraussetzen, rasch an die Grenzen typischer Consumer-Hardware stoßen. Durch Umkonfiguration lässt sich dieses Verhalten jedoch beliebig verändern, wodurch es problemlos möglich ist, mehrere IPFS-Instanzen auch auf günstigen, leistungsschwachen Cloud-Computing-Einheiten zu betreiben. Sybil-Attacken im großen Stil lassen sich dadurch allerdings noch nicht umsetzen. Umgehen lässt sich dieser Umstand, wenn man sich vor Augen führt, dass ein Angriff auf die unteren Schichten des IPFS-Stacks auf Grund eines konzeptionell durchgängig umgesetzten Schichtenmodells einerseits nicht den Betrieb von voll funktionsfähigen IPFS-Instanzen voraussetzt, gleichzeitig aber besonders weitereichende Folgen hätte (siehe Abschnitt 4.2.1).

Ausgehend von dieser Prämisse wurde nach Möglichkeiten gesucht, eine rein auf die elementare Peer-to-Peer-Netzwerkfunktionalität beschränkte Minimalvariante der IPFS-Codebasis zu betreiben. Tatsächlich findet sich der notwendige Programmcode inklusive Dokumentation als Teil des Peer-to-Peer-Moduls von IPFS. Berücksichtigt man zusätzlich die Effizienz der Ed25519-Identifikatorgenerierung im Vergleich zur Standardkonfiguration von IPFS-Knoten (siehe Abschnitt 4.2.3), ergibt sich folgende Situation: Sowohl Erstellen, als auch Betreiben von Instanzen im Rahmen von Sybil-Angriffen ist um ein Vielfaches ressourcenschonender möglich, als der Einsatz „ehrlicher“, vollwertiger IPFS-Knoten. In diesem Zusammenhang ist festzuhalten, dass es sich hierbei um konzeptbedingte Umstände handelt, welche nicht durch einfaches „Nachbessern“ behoben werden können:

1. Die (als zentrale Eigenschaft propagierte) dezentrale Organisation des IPFS-Netzwerks eröffnet Angreifern prinzipbedingt die Möglichkeit, beliebig viele Identifikatoren zu generieren (siehe [9], bzw [17]).

2. Diese offene Architektur ermöglicht es folglich auch, mehrere (sich gezielt disruptiv verhaltende) Instanzen zu betreiben.

3. Würde man von einem dezentralen Modell zu einem zentralen übergehen, hätte dies einen Kompatibilitätsbruch mit allen bestehenden Instanzen zur Folge und würde auch eines der Hauptargrumente für IPFS – Dezentralisierung – entkräften.

4. Selbst wenn man ungeachtet etwaiger Kompatibilitätsbrüche z.B. Proof-of-Work zur Identifikatorgenerierung vorschreiben würde, wäre dies keine Lösung des Problems, sondern würde lediglich den Initialaufwand von Sybil-Attacken erhöhen, was im Sinne einer Fixkostendegression (einmaliges Generieren von Identifikatoren, und diese für beliebig viele

spätere Attacken einsetzen) keine Lösung für das zu Grunde liegende Problem darstellt.

Zusätzlich zur nachweislich effizienten Generierung von ECC-basierten Identifikatoren haben initiale Versuche gezeigt, dass der Betrieb von tausenden, auf die Peer-to-Peer-Funktionalität reduzierten, Instanzen auf einem durchschnittlich leistungsfähigen Computer möglich ist.
Zielt man auf bestimmte weiterführende Attacken ab, lässt sich die Funktionalität von diesen Instanzen weiter beschneiden, was die Effektivität von Angriffen weiter erhöht. Nachfolgend wird der in diesem Abschnitt beschriebene Ansatz für Versuche zu Eclipse-Attacken Zensur weiterverfolgt.

4.2.5. Versuche zu Eclipse-Attacken und Zensur

Wie die ursprüngliche Kademlia-Implementierung bevorzugt IPFS Verbindungen zu anderen Knoten, welche sich über längere Zeit als stabil erwiesen haben. Unter anderem wird dadurch verhindert, dass eine große Zahl bösartiger Knoten betreiben und schlagartig im Netzwerk etabliert werden kann.
Nichtsdestotrotz werden neueingehende Verbindungen akzeptiert, wodurch auf Basis einer wie im vorherigen Abschnitt beschriebenen Sybil-Attacke mittels im Funktionsumfang reduzierter Knoten (im Folgenden als Angriffsknoten bezeichnet) gezielt Verbindungen zu einem Opfer aufgebaut werden können.
In der Ausgangskonfiguration versucht ein IPFS-Knoten zwischen 600 und 900 aktive Verbindungen zu halten. Verbindet man sich (hochgradig parallelisiert) kontinuierlich mit tausenden Angriffsknoten zu einem Knoten hin, kann die Anzahl von Verbindungen zu legitimen Knoten auf unter 200 gedrückt werden. Eine Eclipse-Attacke ist auf diese Weise nicht möglich.
Die IP-Netzwerktopologie und insbesondere Network Address Translation haben jedoch Einfluss auf die Konnektivität von Knoten. Versuche innerhalb desselben Subnetzes, hinter einem Network-Address-Translator, haben (mangels guter Konnektivität des Opfers) zumindest phasenweise zum Erfolg geführt. Jedoch ist auch dies nicht allgemein reproduzierbar und daher nur bedingt aussagekräftig.
Einzig in dem Fall, dass Verbindungen zu einem Opfer, dessen Verbindungsunter- und Obergrenzen von 600, bzw. 900 auf 20 bzw. 40 reduziert wurden, unmittelbar nach dessen Start hergestellt werden, konnten fallweise erfolgreiche Eclipse-Attacken durchgeführt werden. Diese konnten genutzt werden, um dem Opfer den Zugang zum gesamten IPFS-Netzwerk zu verwehren. Versuche mit zusätzlichen Ressourcen, welche auch geografisch auf mehrerer Rechenzentren verteilt Angriffe ausführen (und so ein Botnet approximieren sollen) wurden auf Grund rechtlicher Bedenken nicht durchgeführt.

4.3. Sicherheitsaspekte von Peer-to-Peer-Netzen im Kryptowährungskontext

Moderne (dezentrale) Kryptowährungen setzen (typischerweise unstrukturierte) Peer-to-Peer-Netzwerke ein, um (Transaktions-)Informationen zwischen Teilnehmern auszutauschen. Da es übelicherweise keine zentrale Instanz gibt, welche eine Kryptowährung und deren Transaktionen reglementiert, kann das Peer-to-Peer-Paradigma auf Netzwerkebene als logische Konsequenz des übergeordneten Konzepts betrachtet werden. Die Sicherheit solcher Systeme, welche Transaktionen zwischen Parteien ermöglichen, die einander nicht vertrauen (müssen), ergibt sich durch (je nach Implementierung und Konzept) unterschiedliche Konsensmechanismen. Einen ausführlichen Überblick über die Konzepte solcher Systeme können einer A-SIT-Studie zum Thema Blockchain [18] entnommen werden.

Die Gemeinsamkeit allen Konsensmechanismen ist, dass jede im Netzwerk vertretene Instanz unabhängig vom restlichen Netzwerk die Konformität von Aktionen (üblicherweise Transaktionen) validiert und selbst lediglich konforme Aktivitäten akzeptiert und weiterleitet. Durch einen Mehrheitsentscheid wird festgelegt, welche Aktionen vom Netzwerk als Ganzes als valide angesehen werden – so lange sich die Mehrheit (>50%) des Netzwerks einig ist, bzw. ehrlich verhält, halten alle Sicherheitsgarantien. Dieser Mechanismus fußt auf drei Säulen auf:

1. Ein dezentral fortgeführtes zentrales Transaktionsregister (Blockchain), welches als gemeinsame Datenbasis für alle Netzwerkteilnehmerinnen und Netzwerkteilnehmer dient

2. Ein Resource-Testing-Verfahren, welches festlegt, welche Art von Ressourcen Teilnehmerinnen und Teilnehmer aufbringen müssen, aktiv Daten in Form eines Blocks in das Transaktionsregister aufnehmen zu dürfen

3. Ein Anreizsystem, welches das Erstellen von gültigen Blöcken belohnt

Der durch das Anreizsystem entstehende Wettbewerb um das Erstellen neuer, gültiger Blöcke wird als *Mining* bezeichnet. Durch die unabhängige, effiziente Validierbarkeit der gemeinsamen Datenbasis, auf deren Basis alle Aktionen im Netzwerk ausgeführt werden, sind die (Sicherheits)-anforderungen an das darunterliegende Peer-to-Peer-Netzwerk verhältnismäßig gering. Beispielsweise werden Identifikatoren direkt von IP-Adressen abgeleitet und nehmen lediglich im Rahmen der Lokalisierung von Knoten für die Weiterverbreitung von Transaktionsinformationen und Blöcken eine tragende Rolle ein. Traditionelle Effizienzkriterien für Peer-to-Peer-Netze rücken allein schon deshalb in den Hintergrund, da im Wesentlichen Broadcast-Traffic produziert wird. Dennoch muss ungehinderter Zugriff auf die zentrale Datenbasis möglich sein, da sonst beispielsweise Double-Spending-Angriffe relativ ressourcenschonend durchgeführt werden können.

Konsensverfahren von Kryptowährungen wie Bitcoin arbeiten unter der Annahme, dass alle Teilnehmerinnen und Teilnehmer eine aktuelle, vollständige Sicht auf alle gültigen Blöcke im Netzwerk haben. Wird diese Annahme verletzt, indem Netzwerkknoten beispielsweise von (von miteinander kooperierenden) Angreiferinnen und Angreifern kontrollierten Knoten auf Peer-to-Peer-Ebene in Form einer Eclipse-Attacke abgeschnitten werden, ergibt sich folgende Situation: Angreiferinnen und Angreifer müssen im Fall eines Proof-of-Work-Konsensmechanismus nicht mehr 51% der Gesamtrechenleistung aufbringen, um einen Mehrheitsentscheid zu ihren Gunsten zu manipulieren, sondern lediglich mehr Rechenleistung als der abgeschnittene Knoten – falls dieser überhaupt Mining betreibt. Grund dafür ist, dass das Netzwerk aus Sicht des Opfers lediglich aus sich selbst und den von Angreiferinnen und Angreifern kontrollierten Knoten besteht. Trotzdem muss im Rahmen eines solchen Angriffs genug Rechenleistung aufgebracht werden, um weiterhin gültige Blöcke produzieren zu können.

Ein weiterer Angriffsvektor ergibt sich, wenn beispielsweise einem Händler vorenthalten wird, dass im Rahmen der Bezahlung referenzierte Kryptowährungstoken vom Kunden bereits ausgegeben wurden. Unter anderem um derartige Angriffe zu verhindern, wird empfohlen, vor der Übergabe gekaufter Ware abzuwarten, bis das Netzwerk die entsprechende Transaktion als gültig in einen Block aufnimmt (sicherheitshalber sollen überdies zusätzliche neue Blöcke abgewartet werden). Wird das Netzwerk von Angreiferinnen und Angreifern jedoch derart partitioniert, dass der Händler, sowie ein Teil des Netzwerks, welcher weniger als die Hälfte des Einflusses auf das Konsensverfahren hat, vom Rest des Netzwerks abgeschnitten wird, wird der Händler weiterhin Bestätigungen über die Korrektheit von Transaktionen erhalten. Allerdings handelt es sich dabei um eine beschränkte Sicht auf das Netzwerk und Angreiferinnen und Angreifer können dieselben Token in unabhängigen Transaktionen innerhalb der anderen Netzwerkpartition ausgeben. Zu einem späteren Punkt kann diese Eclipse-Attacke beendet werden und alle von der schwächeren Netzwerkpartition zuvor als valide anerkannten Transaktionen werden von der tatsächlichen Mehrheit des gesamten Netzwerks für ungültig erklärt. Angreiferinnen und Angreifer haben zu diesem Zeitpunkt die erworben Waren und/oder Dienstleistungen bereits erhalten, und sich somit auf Kosten des Händlers bereichern können, ohne selbst am Konsensverfahren teilnehmen zu müssen.

Beschrieben wurden derartige Angriffe bereits vor einigen Jahren in einschlägiger Literatur und 2015 wurde gezeigt, dass das Bitcoin-Netzwerk dafür auch praktisch anfällig ist [19]. Die Kosten für Angriffe waren im Bitcoin-Fall verhältnismäßig gering, da einige bereits seit Jahren bekannte Abwehrmaßnahmen (wie in Abschnitt 4.2.5) von Bitcoin ursprünglich nicht umgesetzt wurden. Insbesondere bevorzugten Netzwerkknoten ursprünglich neu eingehende Verbindungen gegenüber bestehenden. Dadurch fiel es Angreiferinnen und Angreifern leicht, sich in Routingtabellen

gegenüber langlebigen, ehrlichen Teilnehmerinnen und Teilnehmern zu etablieren. Diese konzeptionellen Probleme wurden mittlerweile behoben.
Einen völlig anderen Ansatz verfolgt der im Rahmen dieses Projekts entwickelte Ansatz für sichere Peer-to-Peer-Netze. Dieser wird im nachfolgenden Abschnitt vorgestellt.

5. Peer-to-Peer-Netzwerksicherheit durch Remote-Attestation

In der Literatur zum Thema Sicherheitsmaßnahmen im Peer-to-Peer-Kontext herrscht weitgehend Einigkeit darüber, dass einzig reglementierte Identifikatorgenerierung tatsächlich Abhilfe gegen typische Angriffe wie Sybil- und Eclipse-Attacken schaffen kann. Dies spiegelt sich unter anderem in einer Studie wider, welche über neunzig Konzepte zu diesem Thema kategorisiert und zum ebendiesem Schluss kommt [12]. Bereits im Rahmen des S/Kademlia-Konzepts (das die Basis für viele weitere Arbeiten zu diesem Thema bildet) wurde als Alternative zu Proof-of-Work auch ein klassisches PKI-Szenario für sichere Identifikatorgenerierung angedacht [12].
Strebt man eine dezentrale, offene Organisation eines Peer-to-Peer-Netzwerks an, und will man gleichzeitig ein sicheres System schaffen, ergibt sich jedoch ein Widerspruch der bisher nicht aufzulösen war.

Im Rahmen dieses Projekts konnte dieses Problem nun erstmals insofern gelöst werden, als dass ein allen voran praktisch gangbarer Lösungsweg gefunden wurde. Die Basis bilden die Remote-Attestation-Features aktueller Android-Smartphones. Im Prinzip handelt es sich dabei um die Umsetzung bereits vor Jahrzehnten vorgeschlagener Trusted-Computing-Konzepte [20]. Gemein ist allen Ansätzen aus diesem Umfeld, dass Trusted-Computing nur dann sinnvoll eingesetzt werden kann, wenn die Integrität eines Systems auch aus der Distanz (z.B. über das Internet) nachgewiesen werden kann. Üblicherweise werden Enrolment- und Zertifizierungsverfahren auf Basis von manipulationssicheren, vertrauenswürdigen Hardwaremodulen angewandt, was jedoch im Kontext dezentraler Peer-to-Peer-Netze ebenfalls zu einem Widerspruch führt („Welche Instanz entscheidet wie, wessen Geräte als vertrauenswürdig angesehen werden können?“).

Das ihm Rahmen dieses Projekts entwickelte Konzept umgeht dieses Problem schlicht dadurch, dass es aktuelle Android-Smartphones und deren kryptografische Hardware für die Erstellung von selbstzertifizierenden Identifikatoren (siehe Abschnitt 4.2) heranzieht [21]. Die Remote-Attestation-Verfahren moderner Android-Geräte machen es darüber hinaus auf Basis eines kryptografisch verifizierten Bootvorgangs möglich, festzustellen, ob Betriebssystem und Applikationen, die Attestation entsprechend verwenden, modifiziert wurden. Daraus ergibt sich die Situation, dass der Distributor einer Android-Applikation (in diesem Fall einer Komponente eines Peer-to-Peer-Netzwerks) im Rahmen einer jeden Kommunikation mit dieser Applikation feststellen kann, ob diese auch tatsächlich in unveränderter Form auf einem echten Gerät (und nicht auf einem Emulator) ausgeführt wird.
Im Gegensatz zu Desktop-Betriebssystemen haben Nutzerinnen und Nutzer normalerweise keinen Vollzugriff auf Mobilgeräte, sondern dürfen lediglich die vom Betriebssystem und von Applikationen explizit zur Verfügung gestellten Ressourcen nutzen. In Kombination mit einem verifizierten Bootprozess, dessen Status mittlerweile auch im Rahmen von Remote-Attestation-Verfahren ausgelesen werden kann, wird das gesamte Betriebssystem zu einer Trusted-Computing-Basis. Entscheidend ist in diesem Zusammenhang, dass diese Aussage auf alle neuen Geräte, welche mindestens mit Android in der Version 8.0 ausgeliefert werden, zutrifft, ohne dass diese innerhalb eines Mobile-Device-Management-Systems betrieben werden müssen. Unter der Annahme, dass ein unmodifiziertes Betriebssystem vertrauenswürdig ist, ergibt sich somit die Situation, dass mit zunehmender Verbreitung immer aktuellerer Android-Versionen vertrauenswürdige Geräte für den Betrieb verteilter Anwendungen zur Verfügung stehen.
Im Peer-to-Peer-Kontext ist dies insofern direkt relevant, als das die Android-Remote-Attestation-Prozeduren ohne den Einbezug dritter (wie z.B. eventuell von Google betriebene Services) sondern mittels X.509-Zertifikatsketten und einem Google Root-Zertifikat als Trust-Anchor direkt zwischen Geräten umgesetzt werden können. Konkret wurden diese Eigenschaften aktueller Android-Geräte eingesetzt, um Sybil-Attacken (und in Folge Eclipse-Attacken) zu verhindern und Identifikatoren kryptografisch an Geräte zu binden. Dadurch können nicht mehr beliebig Identifikatoren generiert werden, sondern für den Betrieb eines jeden Peer-to-Peer-Netzwerkknotens ist je ein Android-Gerät

notwendig ist. Die Hauptfunktion wird im Rahmen dieses Konzepts nach wie vor von leistungsstarken, typischerweise breitbandig angebundenen Desktop-Geräten zur Verfügung gestellt, jedoch wird ein Pairing mit einem Android-Smartphone für die Erstellung von selbstzertifizierenden Identifikatoren und die damit einhergehenden Signaturoperationen vorangeschrieben. Konzeptionell übernehmen die im Rahmen dieser Lösung eingesetzten Smartphones damit die Rolle von global als vertrauenswürdig anerkannten Smartcards. Für technische Details dieser Lösung wird auf die Publikation zum Thema [21] verwiesen.

6. Fazit

Im Rahmen dieses Projekts wurden aktuell im Einsatz befindliche Peer-to-Peer-Netze auf ihre Sicherheitskonzepte hin analysiert. Ausgehend von einer allgemeinen Definition von peer-to-peer-spezifischen Angriffen und Sicherheitsproblemen wurde insbesondere der Prozess der Identifikatorgenerierung im Rahmen von IPFS auf potentielle Schwachstellen hin analysiert. Dabei haben sich konzeptionelle Probleme gezeigt, welche prinzipbedingt nicht behoben werden können. Zwar konnten Teilerfolge in Richtung wirksamer Angriffe erreicht werden, verlässlich reproduzierbare Erfolge wurden jedoch nicht erzielt. Nichtsdestotrotz bleibt offen, inwieweit das Sicherheitskonzept auch tatsächlich seinen Anforderungen gerecht wird. Die im Rahmen dieses Projekts durchgeführten Analysen lassen diesbezüglich keine eindeutige Aussage zu, können aber als Basis für komplexere Angriffe auf weiteren Ebenen des IPFS-Stacks herangezogen werden.

Es wurde darüber hinaus ein Überblick über die Sicherheitskonzepte von BitTorrent als Vertreter von Filesharing-Netzwerken gegeben und exemplarisch an Hand von Bitcoin illustriert, wie wichtig ein sicherer Peer-to-Peer-Netzwerklayer ist, auch wenn die eigentlichen Sicherheitsgarantien eines Systems von einer höheren Abstraktionsschicht zur Verfügung gestellt werden. Abschließend wurde ein im Rahmen dieses Projekts entwickeltes neues Sicherheitskonzept für dezentrale Peer-to-Peer-Netzwerk dargelegt, welches in Form einer Wissenschaftlichen Publikation zum Thema auch auf technischerer Ebene ausgearbeitet wurde.

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] J. Benet, D. Dalrymple und N. Greco, „Proof of Replication," 27 07 2017. [Online]. Available: https://filecoin.io/proof-of-replication.pdf. [Zugriff am 31 10 2018].

[3] S. Nakamoto, „Bitcoin: A Peer-to-Peer Electronic Cash System," 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf. [Zugriff am 31 10 2018].

[4] K. Bode, „The Rise of Netflix Competitors Has Pushed Consumers Back Toward Piracy," VICE Media LLC, 08 10 2018. [Online]. Available: https://www.vice.com/en_us/article/d3q45v/bittorrent-usage-increases-netflix-streaming-sites. [Zugriff am 20 05 2019].

[5] 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.

[6] S. Ratnasamy, P. Francis, M. Handley, R. Karp und S. Shenker, „A Scalable Content-Addressable Network," in *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, New York, Ny, USA, ACM, 2001, pp. 161-172.

[7] 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.

[8] K. Pearson, „The Problem of the Random Walk," *Nature,* Bd. 72, Nr. 1865, 1905.

[9] B. Prünster, „Sichere Peer-to-Peer-Netze auf Basis von Selbstzertifizierung," 07 05 2018. [Online]. Available: https://technology.a-sit.at/sichere-peer-to-peer-netze-auf-basis-von-selbstzertifizierung/. [Zugriff am 11 07 2018].

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

[11] 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.

[12] 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.

[13] B. Cohen, „Incentives Build Robustness in BitTorrent,“ 22 05 2003. [Online]. Available: http://www.bittorrent.org/bittorrentecon.pdf. [Zugriff am 16 05 2019].

[14] T. Locher, P. Moore, S. Schmid und R. Wattenhofer, „Free Riding in BitTorrent is Cheap,“ in *5th Workshop on Hot Topics in Networks (HotNets)* , Irvine, California, USA, 2006.

[15] M. Piatek, T. Isdal, T. Anderson, A. Krishnamurthy und A. Venkataramani, „Do incentives build robustness in BitTorrent,“ in *4th USENIX Symposium on Networked Systems Design & Implementation (NSDI)*, USENIX, 2007.

[16] A. Hill, „Adding the next million peers to IPFS,“ Medium, 07 06 2018. [Online]. Available: https://medium.com/textileio/adding-the-next-million-peers-to-ipfs-76d356352d14. [Zugriff am 20 05 2019].

[17] B. Prünster, D. Ziegler, C. Kollmann und B. Suzic, „A Holistic Approach Towards Peer-to-Peer Security and why Proof of Work Won't Do,“ in *14th EAI International Conference on Security and Privacy in Communication Networks*, Singapur, Springer, 2018.

[18] A. Marsalek und B. Prünster, „Technologieüberblick Blockchain,“ 24 10 2016. [Online]. Available: https://technology.a-sit.at/technologieueberblick-blockchain/. [Zugriff am 24 04 2018].

[19] E. Heilman, A. Kendler, A. Zohar und S. Goldberg, „Eclipse Attacks on Bitcoin{\textquoteright}s Peer-to-Peer Network,“ in *24th USENIX Security Symposium (USENIX Security 15)*, Washington, D.C., USENIX Association, 2015, pp. 129-144.

[20] D. o. Defense, „Trusted Computer System Evaluation Criteria“. USA Patent DoD 5200.28-STD, 12 1985.

[21] B. Prünster, E. Fasllija und D. Mocher, „Master of Puppets: Trusting Silicon in the Fight for Practical Security in Fully Decentralised Peer-to-Peer Networks,“ in *Proceedings of the 16th International Security and Cryptography (SECRYPT)*, SciTePress - Science and Technology Publications, 2019.

[22] M. A. Konrath, M. P. Barcello und R. B. Mansilha, „Attacking a Swarm with a Band of Liars: evaluating the impact of attacks on BitTorrent,“ in *Seventh IEEE International Conference on Peer-to-Peer Computing*, Galway, Irland, IEEE, 2007.

[23] B. N. Levine, C. Shields und N. B. Margolin, „A Survey of Solutions to the Sybil Attack,“ Amherst, 2006.