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

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

A-8010 Graz, Inffeldgasse 16a
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

TRAFFIC-ANALYSE MOBILER ANWENDUNGEN

Version 1.0, Mai 2017
Johannes Feichtner – johannes.feichtner@a-sit.at

Zusammenfassung: Unterstützend wurde bei der Inspektion mobiler Anwendungen auch bisher bereits untersucht, welche Daten von Applikationen ins Internet gesendet und von dort empfangen werden. Wurden hierbei Daten im Klartext übertragen, konnten daraus intuitive Rückschlüsse über ihre Verwendung im Kontext einer Mobilanwendung gezogen werden. Durch die zunehmende Verschlüsselung (HTTPS/TLS) von Netzwerkverkehr wird die Aussagekraft von aufgezeichneten Datenpaketen jedoch beschränkt und erlaubt naturgemäß wenig Rückschlüsse über die übertragenen Daten.

Im Zuge dieses Projekts wurde nach Lösungen gesucht, um, unter Anwendung aktueller wissenschaftlicher Verfahren, ein Verständnis über das Verhalten mobiler Anwendungen aus ihren Datenübertragungen abzuleiten. Aufbauend auf bestehenden Ansätzen zur Analyse von Netzwerkverkehr wurde eruiert, ob es möglich wäre, Mobilanwendungen gezielt zu identifizieren, selbst wenn Verschlüsselung den Zugriff auf die übertragenen Inhalte unmöglich macht. Unter Fokussierung auf Metadaten von Übertragungen wurde ein Ansatz gesucht, um Applikationen individuell voneinander zu unterscheiden.

Eine praktische Implementierung des vorgeschlagenen Analysekonzepts wurde auf einen mitgeschnittenen Datensatz angewandt. Dieses Testszenario hat ergeben, dass sich Mobilanwendungen anhand ihres Netzwerkverkehrs mit einer Zuverlässigkeit von 83,3% klassifizieren ließen – ungeachtet dessen ob Kommunikation in verschlüsselter oder unverschlüsselter Form vorlag.

Dieses Dokument beschreibt die Ergebnisse der vorgenommenen Forschung, stellt das erarbeitete Konzept vor und erörtert die technischen Analyse-Möglichkeiten. Die aus diesem Projekt gewonnenen Ergebnisse dienen als Wissensbasis für ähnliche Untersuchungen im Netzwerkbereich und liefern anderseits aufschlussreiche Informationen über den Netzwerkverkehr mobiler Anwendungen.

Inhaltsverzeichnis

1. Einleitung

Das Mitschneiden und die Analyse von Netzwerkverkehr ist seit mehreren Jahren eine gängige Praxis um relevante Informationen aus dem Datenfluss zu extrahieren. Neben ausgereiften Systemen zur Einbruchserkennung (*Intrusion Detection*), findet das Konzept auch immer häufiger Einzug in Firewalls, die von Privatpersonen betrieben werden. Eingesetzt zur Identifikation und Klassifikation von Anomalien in Übertragungen, können ausgereifte Ansätze der Traffic-Analyse Einbrüche und Angriffe auf Netzwerke frühzeitig erkennen. Neben dem möglichen Schutzzweck ist die Inspektion von Datenverkehr auch dazu geeignet, als Informationsquelle für Analysen auf einer Metaebene zu dienen. Neben dem eigentlichen Inhalt einer Übertragung können Parameter wie etwa die Häufigkeit, das Volumen und die Verteilung von Datenpaketen relevant sein, um Verhaltensmuster zu erkennen und korrekterweise zu klassifizieren.

Bei der Untersuchung von Anwendungen für Mobilplattformen spielt aufgezeichneter Datenverkehr auch eine wesentliche Rolle, um ein Verständnis für das Verhalten einer Applikation zu gewinnen. Neben der zentralen Frage, welche Daten Applikationen eigentlich ins Internet übertragen, gilt es bei einer Inspektion ebenfalls festzustellen, *wie* die Übertragung stattfindet. Für die Vertraulichkeit persönlicher Daten ist es essentiell, dass deren Transfer über TLS-geschützte Verbindungen stattfindet. Eine häufig eingesetzte Möglichkeit um dies zu eruieren, ist die Durchführung eines sog. „Man-in-the-Middle“ (MITM) Angriffes. Der/die Ausführende nimmt dabei eine Position zwischen dem sendenden Mobilgerät und einem bzw. mehreren Zielservern ein und schneidet den Datenverkehr mit (*Packet Dump*). Eine manuelle Untersuchung des Resultats, etwa mithilfe des Tools Wireshark[1], ermöglicht schließlich einen Einblick in übertragene Daten.

Da die Verwendung von TLS für die Vertraulichkeit einer Übertragung von wesentlicher Bedeutung ist, animieren Google und Apple die Entwickler von Mobilanwendungen, entsprechende Schutzmechanismen zu aktivieren. Mit dem Erscheinen von iOS in Version 9, hat Apple den Mechanismus der „App Transport Security“[2] eingeführt, welcher vorschreibt, dass Netzwerkverbindungen prinzipiell über HTTPS geschützt sein müssen. Analog dazu setzt auch Google Initiativen[3][4] um, die Entwickler zum Einsatz von HTTPS-Verbindungen veranlassen sollen. Der zunehmende Einsatz von TLS in Mobilanwendungen trägt positiv zur Vertraulichkeit von Verbindungen bei und ist aus dieser Perspektive unzweifelhaft zu begrüßen. In Bezug auf MITM-Angriffe bedeutet er jedoch, dass die Inhalte von mitgeschnittenem Datentransfer nicht mehr lesbar sind und eine etwaige Analyse lediglich auf Metadaten beruhen kann. Obwohl ein MITM-Angriff auf TLS-Verbindungen prinzipiell ebenfalls durchführbar ist, steigt bei korrekter Prüfung eines Server-Zertifikats in einer Mobilanwendung sowie bei Einsatz von „Certificate Pinning“ der nötige Analyse-Aufwand überdurchschnittlich und ist nicht universell für jede Applikation anwendbar.

1.1. Ziele dieses Projekts

Im Zuge dieses Projekts wird nach Möglichkeiten gesucht, Rückschlüsse auf die verwendeten Mobilanwendungen zu ziehen, selbst wenn TLS zur Verbindungssicherung eingesetzt wird. Da der Schutzmechanismus von TLS keinen Zugriff auf die übertragenen Inhalte ermöglicht, kann sich die Analyse lediglich auf die zur Übertragung gehörenden Metadaten stützen. Mit der Intention, trotz TLS den Traffic von Mobilanwendungen zu analysieren, ergeben sich mehrere wissenschaftliche Fragestellungen, die für dieses Projekt relevant sind:

- **Ist es grundsätzlich möglich, trotz TLS-Verschlüsselung relevante Informationen aus dem Datenverkehr zu extrahieren?**
  Die in Firewalls oder bei Intrusion Detection zur Anwendung kommenden Analyse-Ansätze sind häufig darauf angewiesen, Zugriff auf Klartext-Inhalte einer Übertragung zu haben[5]. Ist dies gegeben, kann etwa nach vordefinierten Mustern (Patterns) gesucht werden, deren Vorkommen als problematisch erkannt werden könnte.

[1] https://www.wireshark.org
[2] https://www.apple.com/business/docs/iOS_Security_Guide.pdf
[3] https://developer.android.com/training/articles/security-ssl.html
[4] https://android-developers.googleblog.com/2017/04/android-o-to-drop-insecure-tls-version.html
[5] https://www.heise.de/security/meldung/US-CERT-warnt-vor-HTTPS-Inspektion-3660610.html

Im Hinblick auf den Mobilbereich bedeutet dies, dass bei der manuellen Analyse von mitgeschnittenem Datenverkehr keine Aussage mehr darüber getroffen werden kann, ob eine Übertragung persönliche Daten einschließt.

Die wesentliche Frage ist somit, welche Metadaten für eine Traffic-Analyse verbleiben und ob sie hinreichend aussagekräftig sind um damit Rückschlüsse auf das Verhalten von Applikationen zu liefern.

- **Welche Eigenschaften charakterisieren einzelne Mobilanwendungen?**
  Die bei einem MITM-Angriff auf ein Mobilgerät mitgeschnittenen Daten unterscheiden nicht zwischen einzelnen Anwendungen. Der Packet Dump, typischerweise im PCAP-Format[6] vorliegend, ist chronologisch strukturiert und listet alle Pakete auf, die während des Mitschnitts vom Gerät empfangen oder gesendet werden. Für die zielgerichtete Traffic-Analyse einzelner Applikationen ist die Gesamtdarstellung jedoch nicht hilfreich.

  Die Frage ist folglich, ob bzw. anhand welcher Attribute in Datenpaketen sich einzelne Applikationen voneinander unterscheiden.

- **Auf welcher Verbindungsebene ist eine Analyse sinnvoll?**
  TLS-Pakete sind gekapselt in darunterliegenden Netzwerkschichten, die ihrerseits jeweils unterschiedliche Aufgaben erfüllen und sich auch in den Metadaten unterscheiden. Welche dieser Daten dabei einzelne Anwendungen besonders ausdrücklich charakterisieren, gilt es herauszufinden.

Mit einem Ziel eine Wissensbasis für vergleichbare Analysen aufzubauen, wird in diesem Projekt der Fokus auf die angeführten Kernfragen gelegt.

2. Gegenstand der Traffic-Analyse

In der Perspektive der Netzwerkschichten nach dem OSI-Modell baut TLS unmittelbar auf TCP auf. Wie in Abbildung 1 illustriert, kann die Traffic-Analyse von verschlüsseltem Datenverkehr verschiedene Schichten mit einbeziehen, muss jedoch ohne die Inhalte der Applikationsprotokolle (grün unterlegt) auskommen. Die noch „auswertbaren“ Daten beschränken sich somit auf die Sitzungsebene (TLS), die Transportschicht (TCP), die Netzwerkschicht (IPv4 / IPv6). Darunterliegende Schichten adressieren Pakete nur zwischen Netzwerkinterfaces bzw. nicht global gerouteten Netzen und werden daher nicht weiter in Betracht gezogen.

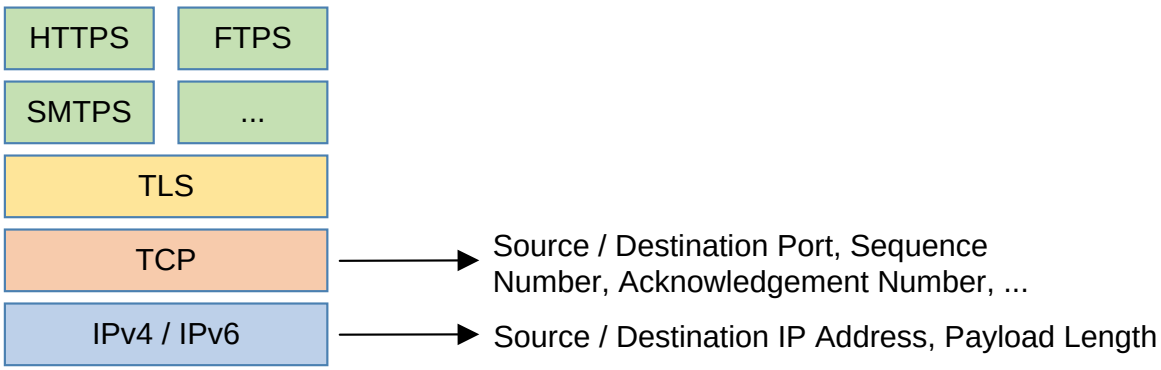

Abbildung 1. „Lesbare“ Daten bei TLS-Verschlüsselung.

Das TLS-Paket wird bei der Übertragung in einem TCP-Paket gekapselt bzw. jenes wiederum einem IP-Paket. Durch die Betrachtung aller Schichten ergibt sich die Information, von/an welche IP-Adresse etwas gesendet/empfangen wurde, wie groß der gesendete/empfangene Payload ist und mit welchem Dienst kommuniziert wurde. Letzteres ist anhand der Portnummer[7] feststellbar. Einzelne Pakete lassen sich somit relativ unzweifelhaft identifizieren, also dem designierten Dienst

[6] https://wiki.wireshark.org/Development/LibpcapFileFormat
[7] https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

zuweisen. Für die weitere Analyse nehmen wir an, dass Mobilanwendungen von den offiziellen Portzuweisungen nicht abweichen und mit Diensten dem Port entsprechend kommunizieren. Da Mobilanwendungen überwiegend via HTTPS (Port 443) und HTTP (Port 80) Verbindungen unterhalten, muss für die vollständige Traffic-Analyse eine Ebene gefunden werden, die beiden Protokollen inhärent ist. Um also Rückschlüsse auf den Datenverkehr beider Ports ziehen zu können, kann nur auf jener obersten Netzwerkschicht aufgebaut werden, die beiden Protokollen zugrunde liegt. Im konkreten Fall ist dies die Transportschicht (TCP).

Die Information welcher Hostname bei einer Verbindung kontaktiert wird, ist ohne Kenntnis der (durch TLS-verschlüsselten) HTTP-Anfrage nicht erkennbar, da IP-Pakete lediglich die IP-Adresse des Zielservers ausweisen. Praktisch bedeutet das, dass sich so zwar herausfinden lässt, welche IP-Adressen kontaktiert wurden, nicht jedoch welche Domain-Namen das Ziel der Anfrage waren. Dieser Umstand wird dann zu einem Problem, wenn Domains über mehrere IP-Adressen erreichbar sind (= mehrere A-Records in der DNS-Zone). Verbindungen zu diesen Domains können bei der Analyse nicht in Zusammenhang gebracht werden, obwohl sie gewissermaßen „verwandt“ sind. Um somit dennoch eine semantische Relation in den Daten abzubilden, können wir als Hilfsmaßnahme auf den „Reverse Hostname“ einer IP-Adresse zurückgreifen. Die Analyse wird insofern verfeinert indem semantische Verbindungen zwischen einzelnen Verbindungen abgebildet werden.

Das TCP-Protokoll sieht für den Aufbau von Verbindungen einen Handshake und für den Abbau ein „Teardown“-Verfahren vor[8]. Über die dabei ausgetauschten Sequence und Acknowledgement Numbers können einzelne Pakete einer TCP-Session zugeordnet werden. Für unsere Analyse bedeutet das, dass wir nicht notwendigerweise mit einzelnen Datenpaketen arbeiten müssen (deren Payload bei TLS-Verschlüsselung ohnehin nicht lesbar wäre), sondern einzelne TCP-Sessions aus einem Packet Dump extrahieren können. Aus einer vollständigen Sitzung lässt sich außerdem eruieren, welche Datenmenge im Zuge einer Sitzung gesendet und empfangen wurde. Dies lässt per se zwar noch keine Rückschlüsse auf den Inhalt der Übertragung zu, kann aber zugleich schon als Anhaltspunkt dafür dienen, um welche Art der Daten es sich handeln dürfte (Text oder Dateien).

Es lässt sich somit festhalten, dass eine Analyse auf Basis von TCP-Sessions als oberste Ebene eine Analyse ermöglicht, die sowohl HTTPS- als auch HTTP-Traffic berücksichtigt und zudem die Notwendigkeit entfernt, Informationen aus einzelnen Paketen zu extrahieren.

3. Konzeption einer Analyse-Umgebung

Basierend auf den zuvor eruierten Rahmenbedingungen, wird nachfolgend ein Ablauf vorgeschlagen, um den Datenverkehr von Mobilanwendungen zu identifizieren und analysieren.

3.1. Traffic-Aufzeichnung am Mobilgerät

Wie eingangs beschrieben, ist eine wesentliche Herausforderung die Zuordnung von Traffic zu einzelnen Applikationen. Ein klassischer MITM-Angriff liefert einen Packet Dump mit sämtlichen Verbindungen, die in einem gewissen Zeitraum stattgefunden haben. Für Rückschlüsse auf die einzelnen Anwendungen muss jedoch ein Mechanismus gefunden werden, um Verbindungen einer Applikation gezielt zuordnen zu können. Eine externe Aufzeichnung kann diese Daten nicht liefern.

Die im Zuge dieses Projekts durchgeführte Recherche hat hervorgebracht, dass bei Linux-basierten Systemen wie Android sämtliche Internet-Verbindungen auch im Dateisystem abgebildet werden. Konkret heißt das, dass das Betriebssystem bei Aufbau einer Verbindung jeweils die IP-Adressen und Ports des Zielservers und der Quelle sowie die die ausführende User-ID (UID), je nach Transport-Protokoll in den Dateien */proc/net/icmp*, */proc/net/icmp6*, */proc/net/tcp*, */proc/net/tcp6*, */proc/net/udp* oder */proc/net/udp6* vermerken. Diese Informationen sind bei Android von jeder Applikation ohne höhere Benutzerrechte lesbar (kein root-Zugriff notwendig). Dies ermöglicht es festzustellen, zu welcher Anwendung eine Verbindung gehört. Die Folgen für die Analyse sind weitreichend: In einem Packet Dump gesammelter Traffic kann assoziiert werden mit einzelnen

[8] https://tools.ietf.org/html/rfc793

Applikationen. Wir erhalten dadurch Traffic-Muster, die individuell sind pro Applikation und können so gezielt Verbindungsinformationen auswerten.

Um Verbindungen und zugehörige Applikationen gleichzeitig mitzuschneiden, ist eine entsprechende Anwendung am Mobilgerät vonnöten. Eine Recherche hat ergeben, dass es unter Android mehrere Anwendungen gibt, die die Verbindungsinformation der */proc/net* Dateien visuell aufbereiten[9][10]. Für die nachfolgende Analyse des Traffic wird jedoch darüber hinaus der Datenverkehr als Packet Dump benötigt.

3.1.1. NetGuard

Um Datenverkehr unter Android direkt am Gerät (ohne höhere Systemrechte) aufzuzeichnen, kann die VPN-Funktionalität des Geräts zweckentfremdet werden. Seit Android in Version 5 ist es möglich, eigene VPN-Clients zu implementieren und dem System über die *VPNService*-Schnittstelle[11] bereitzustellen. Die Anwendung NetGuard[12] bedient sich dieses Mechanismus um einen fiktiven VPN-Client zu implementieren, der Daten mitschneidet und dann gewöhnlich (ohne VPN-Server) ins Internet weiterleitet.

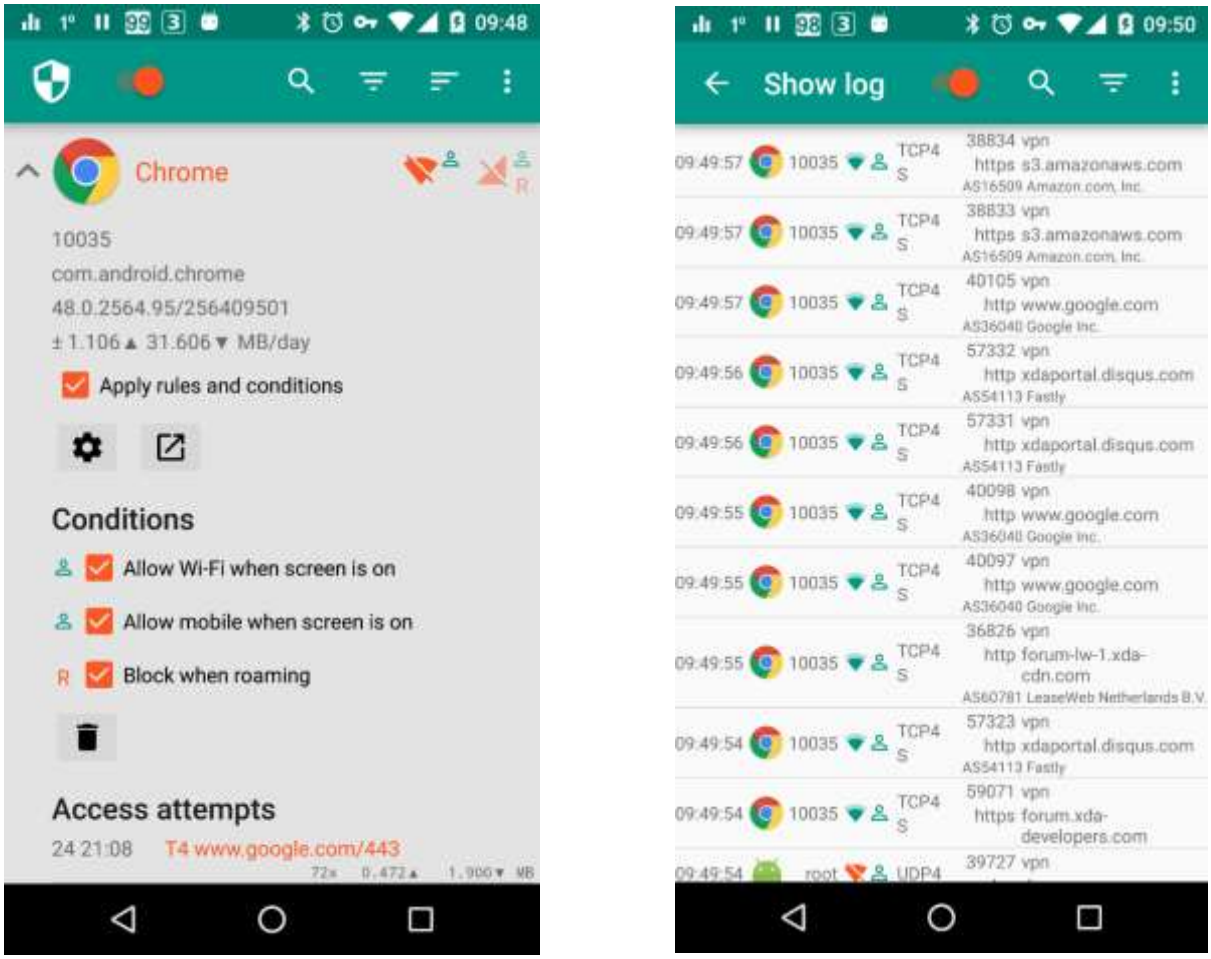

Abbildung 2. Traffic-Mitschnitt am Gerät mithilfe einer fiktiven VPN-Verbindung (NetGuard).

Obwohl die eigentliche Intention der Open-Source-Anwendung die Reglementierung des Internetzugriffes einzelner Anwendungen ist, implementiert sie auch Funktionalität um Datenfluss aufzuzeichnen. Wie in Abbildung 2 illustriert, findet dabei ein chronologischer Mitschnitt statt, der in weiterer Folge im PCAP-Format gespeichert werden kann. Obwohl die Darstellung suggeriert, dass

[9] https://haystack.mobi
[10] https://play.google.com/store/apps/details?id=com.radioopt.netstats
[11] https://developer.android.com/reference/android/net/VpnService.html
[12] https://www.netguard.me

auch der Name der aufgerufenen Domain mitprotokolliert werden würde, entspricht der Wert lediglich dem „Reverse Hostname“ der IP-Adresse. Wie im vorherigen Abschnitt erläutert, ist die eigentlich aufgerufene Domain bei TLS-verschlüsselten Paketen nicht auslesbar.

Für die Rahmen dieses Projekts durchgeführte Traffic-Analyse wird NetGuard erweitert:

- Die im PCAP-Format mitgeschnittenen Datenpakete werden ohne zugehörige Ethernet-Header aufgezeichnet. Obwohl diese Verbindungsebene nicht relevant ist für unsere Analyse, werden die Header benötigt um die resultierenden PCAP-Dateien in weiterer Folge verarbeiten zu können. Da dem VPN-Client unter Android ein Zugriff auf diese Netzwerkschicht nicht gestattet ist, wird stattdessen vor jedes Paket ein Pseudoheader eingefügt, der den gleichen Zweck erfüllt.

- Obwohl NetGuard bereits die Funktionalität mitbringt, Traffic einzelner Anwendungen mitzuschneiden, spiegelt sich diese Assoziation nicht in Packet Dumps wider. Konkret bedeutet das, dass am Mobilgerät zwar visuell erkennbar ist, welche Applikation eine Verbindung aufbaut, diese Information aber nicht in den Packet Dumps vermerkt wird. Die Ursache liegt im PCAP-Format an sich: Eine Hinterlegung von Metadaten (wie die UID einer Mobilanwendung) oder weiteren Informationen ist schlichtweg nicht vorgesehen.

  Um die UID von Anwendungen dennoch mit einzelnen Netzwerkverbindungen zusammenführen zu können, muss eine zweite Aufzeichnung separat geführt werden. Hierbei wird die UID gemeinsam mit einem Zeitstempel vermerkt, der sich seinerseits im PCAP Packet Dump wiederfindet und so bei der späteren Analyse assoziiert werden kann.

Als Resultat dieses Schrittes wird eine Lösung geschaffen, die Datenverkehr am Mobilgerät aufzeichnet, ihn im PCAP-Format ausgibt und darüber hinaus protokolliert, welche Anwendung bei welchem Zeitstempel eine Verbindung unterhalten hat. Einer weiteren Traffic-Analyse wird hierdurch ermöglicht, gezielt Verbindungen und Applikationen miteinander zu assoziieren und so rohe Datenpakete einzelner Mobilanwendungen zu inspizieren.

3.2. Analyse des Datenverkehrs am PC

Der im vorherigen Schritt aufgezeichnete Netzwerkverkehr im PCAP-Format soll im Folgenden weiter untersucht werden. Wie in Abschnitt 1.1 erläutert, soll dabei auf die Fragen eingegangen werden, ob trotz TLS-Verschlüsselung Informationen aus dem Verkehr extrahierbar sind und ob die Übertragungen hinreichend individuell sind um schließlich (bei beliebigem Traffic) vom Datenverkehr auf die verursachende Applikation schließen zu können. Das Ziel dieser Analyse ist somit nicht, spezifischen Datenverkehr zu identifizieren, sondern ihn zu klassifizieren.

3.2.1. Klassifikation von Netzwerkverkehr

Für eine Vielzahl von Anwendungszwecken (Firewalls, Intrusion Detection, Traffic Shaping, Policy Enforcement, Lawful Interception, etc.) ist die Klassifikation und das Verständnis von Datenverkehr ein essentielles Erfordernis. Die meisten Ansätze, die in den letzten Jahren für diesen Zweck entwickelt wurden, basieren auf Methoden der Statistik oder maschinellem Lernen (ML).

Die Studie von Valenti et al. [1] präsentiert eine Taxonomie möglicher Klassifikationstechniken („Classifier“) auf Basis unterschiedlicher Ansätze. Wie aus Abbildung 3 hervorgeht, unterscheiden sich alle Ansätze in ihrer Granularität, der aufzubringenden Rechenleistung und unterliegen zudem einer zeitlichen Komponente.

„Coarse grained“-Ansätze zielen darauf ab, Familien von Protokollen zu erkennen (HTTP vs. Streaming), wohingegen „fine grained“-Algorithmen eine genauere Unterscheidung zwischen einzelnen Protokollen oder dahinterstehenden Anwendungen ermöglichen. Je nach verwendetem Algorithmus muss eine unterschiedliche Anzahl an Paketen im Speicher gehalten und analysiert werden.

Approach	Properties exploited	Granularity	Timeliness	Comput. Cost
Port-based	Transport-layer port [49,50,53]	Fine grained	First Packet	Lightweight
Deep Packet Inspection	Signatures in payload [44,50,60]	Fine grained	First payload	Moderate, access to packet payload
Stochastic Packet Inspection	Statistical properties of payload [26,30,37]	Fine grained	After a few packets	High, eventual access to payload of many packets
Statistical	Flow-level properties [38,45,50,58]	Coarse grained	After flow termination	Lightweight
	Packet-level properties [8,15]	Fine grained	After few packets	Lightweight
Behavioral	Host-level properties [35,36,67]	Coarse grained	After flow termination	Lightweight
	Endpoint rate [7,28]	Fine grained	After a few seconds	Lightweight

Abbildung 3. Techniken zur Klassifikation von Traffic nach [1].

Alle angeführten Ansätze basieren darauf, gewisse Muster („Protokoll-Signaturen“ oder „Patterns“) aus den verwendeten IP-Adress- und Port-Kombinationen zu erkennen oder direkt aus den übertragenen Daten („Payloads“) abzuleiten. Dabei unterliegen sie jedoch teilweise Einschränkungen (siehe auch [2] und [3]):

- Wenn der Payload eines Datenpakets nicht lesbar ist, da etwa TLS-Verschlüsselung zur Anwendung kommt, können keine Patterns in diesen Daten festgestellt werden.
- Der Classifier muss mit der Syntax eines jeden zu erkennenden Payloads vertraut sein.
- Klassifikation auf Basis von IP-Adresse und Port erzielt im besten Fall eine Genauigkeit von 70% wenn Dienste anhand ihres offiziell zugewiesen Ports klassifiziert werden [4].

Im Hinblick auf den in diesem Projekt gegebenen Mobilbereich liegt überwiegend HTTP und HTTPS-Traffic vor. Eine Klassifikation über die verwendeten Ports („*Port-based classification*“) wäre somit nicht zielführend. Wird alternativ dazu Traffic klassifiziert auf Basis des übertragenen Payloads („*Payload-based classification*“), würde dieser Ansatz lediglich für jene Datenpakete funktionieren, die unverschlüsselt übertragen werden. Es sind somit auch keine Ansätze von „Deep Packet Inspection (DPI)“ oder „Stoachstic Packet Inspection (SPI)“ anwendbar, die Zugriff auf Klartextdaten benötigen. Statistische Klassifikationsalgorithmen („Statistical classifiers“) verwenden als Features Angaben zum Datenfluss und würden daher auch mit verschlüsseltem Traffic funktionieren. In diesen Bereich fallen auch sog. „unsupervised“ und „supervised“-Klassifikationsalgorithmen, die Techniken des maschinellen Lernens verwenden. Während der „unsupervised“-Ansatz Daten nach ihrer Heterogenität in Cluster aufspalten kann (z.B. über das kMeans-Verfahren), basiert der „supervised“-Ansatz darauf, dass ein neuronales Netz zuerst mit Trainingsdaten unter Zuhilfenahme entsprechender Verfahren (z.B. Naive Bayes, C4.5 oder Support Vector Machines) angelernt wird. Verhaltensgestützte Algorithmen *(„Behavioural classifiers“)* zielen darauf auf, Applikationen auf dem Zielhost zu erkennen, indem sie Muster aus dem Traffic ableiten, wie beispielsweise die Angabe wie viele Hosts kontaktiert wurden, über welches Transportprotokoll oder über die Anzahl verschiedener Ports. Die Idee dahinter ist, dass verschiedene Anwendungen unterschiedliche Verhaltensmuster generieren. Ein P2P-Programm würde beispielsweise viele Endpunkte auf verschiedenen Ports gleichzeitig kontaktieren, wohingegen ein Webserver üblicherweise auf den Ports 80 und 443 hört.

Für dieses Projekt lassen sich somit folgende Feststellungen treffen:

- Um auch mit TLS-Verbindungen umgehen zu können, sind nur statistische Verfahren bzw. Machine Learning („supervised“ oder „unsupervised“) Techniken oder verhaltensgeschützte Klassifikationsalgorithmen anwendbar.

- Auf Basis der von NetGuard ausgegebenen Daten liegen sowohl Packet Dumps im PCAP-Format als auch „Zuordnungen“ (Labels) zu einzelnen Applikationen vor. Weitere Metadaten können bzw. sollten abgeleitet und in die Analyse einbezogen werden um semantische Relationen zwischen Verbindungen möglichst deutlich abzubilden. Dadurch können letztlich auch Verhaltensmuster von Applikationen hervorgehen, die für eine eindeutige Klassifikation förderlich sein können.

- Durch die „Labels“ und die Generierung weiterer Metadaten (z.B. „Reverse DNS“-Name bei IP-Adressen) drängt sich die Verwendung eines „Supervised Learning“-Ansatzes auf, der auch mit heterogenen Inputdaten (IP-Adressen, Ports, Anzahl an gesendeter und empfangener Bytes einer TCP-Session, etc.) funktioniert.

Zwei Ansätze um Zusammenhänge in PCAP Dumps vor der Verarbeitung mit einem Machine Learning Classifier semantisch korrekt abzubilden, sind „Latent Semantic Indexing“ [5] und „Semantic Patterns Transformation“ [6]. Anlässlich der unmittelbaren Möglichkeit beliebige symbolische (Strings, Bytes) Daten in Relation zu setzen mit numerischen (Anzahl Bytes) verwenden wir für unsere Analyse den letztgenannten Ansatz von Teufl et al.

3.3. Aufbereitung der Daten („Preprocessing“)

Bevor die Klassifikation stattfinden kann, muss der von NetGuard ausgegebene Packet Dump entsprechend aufbereitet und ggf. ergänzt werden. Die dabei ausgeführten Schritte werden im Nachfolgenden detaillierter erklärt.

3.3.1. Zusammenführen von TCP-Session und Mobilanwendung

Um eine Analyse auf einer abstrakteren Ebene durchführen zu können, werden die Pakete zunächst in TCP-Sessions zusammengefasst. Hierzu kommt das bestehende Werkzeug SplitPcap[13] zum Einsatz. Um schließlich in Erfahrung zu bringen, welche Anzahl an Daten innerhalb einer Sitzung mit einem Zielserver hoch- und runtergeladen wurden, wird ermittelt, wie groß der Payload in jedem einzelnen TCP-Paket einer Session ist. Die Bytes des jeweiligen Payloads (zu einer ver- oder unverschlüsselten Verbindung gehörig) werden schließlich in einem Histogramm dargestellt, das ebenfalls der Charakterisierung einer Verbindung dienlich sein könnte. Bei unverschlüsselter Verbindung und der Übertragung von Klartext, würde das Histogramm beispielsweise verstärkt Bytes im ASCII-Bereich (A-Z, a-z, 0-9 und Sonderzeichen) ausweisen.

Indem NetGuard instruiert wurde, zusätzlich zum Packet Dump auszugeben, welche Applikation zu einem gewissen Zeitpunkt eine Verbindung zu einem Ziel geöffnet hatte, kann ein „Matching“ durchgeführt werden. Konkret heißt das, jede TCP-Session kann einer Anwendung spezifisch zugewiesen werden.

3.3.2. Steigerung der Datenqualität durch DNS-Matching

Wie zuvor beschrieben, kann bei verschlüsselten Verbindungen nicht auf den Domainnamen geschlossen werden, den eine Applikation tatsächlich kontaktiert. Ein Ansatz um diesem Umstand beizukommen, wäre die Verwendung des „Reverse DNS“-Hostnamen[14] einer IP-Adresse. Wird ein Paket z.B. an die IP-Adresse 193.170.141.229 gesendet, würde ein rDNS-Lookup ergeben, dass die Adresse mit dem Hostname *cache.google.com* assoziiert ist. Diese Generierung zusätzlicher Metadaten für die Analyse beeinflusst auch die Qualität der Traffic-Daten:

- Vorteilhaft ist, dass eine IP-Adresse so in eine semantische Relation mit einem Hostname gebracht werden kann, der an sich aus den Metadaten in Paketen nicht lesbar ist. Dies kann die Zielgenauigkeit der Traffic-Analyse unterstützen. Findet sich im gleichen „Packet Dump“ beispielsweise auch Datenverkehr mit der IP-Adresse 193.170.141.249, lässt sich eine

[13] https://www.netresec.com/?page=SplitCap
[14] https://tools.ietf.org/html/rfc1912

Relation herstellen zwischen 193.170.141.229 und 193.170.141.249, da beide den gleichen rDNS-Namen ausweisen.

- Der Nachteil ist wiederum, dass es die Analyse verzerrt, da der rDNS-Hostname nicht notwendigerweise jenem Domainnamen entspricht, den eine Applikation auch tatsächlich aufgerufen hat. Beispielsweise verweist die Domain *google.at* u.a. auf die IP-Adresse 193.170.141.229, die Domain *youtube.com* u.a. auf 193.170.141.249. Ein rDNS-Lookup würde diese Information nicht liefern (können), da der „Reverse DNS"-Hostname beider IP-Adressen *cache.google.at* lautet.

Um nun falsche semantische Assoziationen möglichst zu vermeiden und dem realen Verhalten von Anwendungen zu entsprechen, wurde folgende Strategie überlegt: Bevor eine Mobilanwendung überhaupt eine IP-Adresse kontaktieren kann (sofern sie das nicht explizit spezifiziert), muss der in der Applikation hinterlegte Domainname durch den DNS-Client des Betriebssystems aufgelöst werden. Dies hat zur Folge, dass auf Port 53 (UDP) eine DNS-Anfrage an den zuständigen DNS-Resolver gestellt wird. Die Antwort enthält schließlich eine Liste aller IP-Adressen unter denen eine Domain erreichbar ist. Diese DNS-Anfragen werden ebenfalls von NetGuard im Packet Dump vermerkt, können aber nicht mit einer Applikation unmittelbar assoziiert werden, da sie der DNS-Client des Betriebssystems ausführt. Die Idee ist folglich, ausgehend von einer TCP-Session nach dem chronologisch am nächsten liegenden Paket mit einer DNS-Antwort zu suchen, die die IP-Adresse einer Session zurückgibt. Die Beachtung der chronologischen Abfolge ist vor allem deshalb relevant, da eine eben mehrere Domains („Virtual Hosts") bedienen kann, die unter Umständen im gleichen Datenfluss vorkommen.

Anhand eines konkreten Beispiels lässt sich das DNS-Matching und die Relevanz der zeitlichen Abfolge gut darstellen:

Timestamp	Datenpaket
0	DNS-Antwort für Anfrage *www.googleapis.com*: 172.217.22.106
1	TCP-Session mit 216.58.206.5
2	TCP-Session mit 172.217.22.106
3	DNS-Antwort für Anfrage *chromereader-pa.googleapis.com*: 172.217.22.106
4	TCP-Session mit 172.217.22.106

Für die TCP-Session bei Timestamp 1 findet sich womöglich keine DNS-Antwort im Packet Dump. Dies kann entweder daran liegen, dass eine Mobilanwendung die IP-Adresse explizit aufgerufen hat oder, dass der DNS-Client des Betriebssystems den DNS-Eintrag noch von einer früheren Anfrage im Cache hält. Bei unserer Analyse müssen wir somit auf den Reverse Hostname zurückgreifen, welcher *fra16s20-in-f5.1e100.net* lautet. Dass die Applikation eigentlich *inbox.google.com* aufgerufen hat, ist nicht auslesbar.

Für andere TCP-Sessions kann der Domainname jedoch mitunter korrigiert werden. Findet sich beispielsweise vor der Session bei Timestamp 2 eine DNS-Antwort im Dump, die 172.217.22.106 mit einem Hostname assoziiert (Timestamp 0), so kann als Domainname unzweifelhaft der Hostname der DNS-Antwort angenommen werden.

Bei Timestamp 4 findet neuerlich eine Session mit der gleichen IP-Adresse statt. Für unsere Analyse nehmen wir an, dass die davor stattgefundene DNS-Antwort in Relation zur TCP-Session steht und die in der Anwendung aufgerufene Domain *chromereader-pa.googleapis.com* ist. In der Theorie könnte die Anfrage durch die gleiche IP-Adresse auch an *www.googleapis.com* gerichtet sein. In diesem Fall wäre aber die DNS-Antwort bei Timestamp 3 obsolet, da *www.googleapis.com* bereits zuvor aufgelöst wurde. Da wir in den Payload von verschlüsselten Paketen jedoch nicht hineinsehen, lässt sich die theoretische Wahrscheinlichkeit nicht vollständig ausschließen, dass die TCP-Session bei Timestamp 4 (ungeachtet der DNS-Antwort bei Timestamp 3) an erstere Domain gerichtet ist.

Die Qualität der Analyse-Daten wird somit insofern gesteigert, da Domainnamen typischerweise statisch in Mobilanwendungen hinterlegt werden und somit hinreichend gut zur Unterscheidbarkeit

von Anwendungen bei der Klassifikation beitragen sollten. Dieser Effekt tritt insbesondere dann positiv in Erscheinung, wenn Domains hinter Clouddiensten angeboten werden (z.B. Amazon AWS) oder Load Balancing (z.B. Google) verwendet wird. In diesen Fällen ist die IP-Adressen noch praktisch nicht an Domainnamen gebunden und die Verwendung des Reverse Hostnames würde keinen Rückschluss auf die von Mobilanwendungen verwendeten Domains ermöglichen.

3.3.3. Geolocation und IP-Bereich

Im Gegensatz zu Domainnamen enthalten IP-Adressen per se keine Informationen über einen geographischen Bezugspunkt. Bei einer IP-Adresse ist außerdem nicht direkt erkennbar, welchem Bereich sie angehört und ob es eine Relation mit IP-Adressen im gleichen Adressbereich gibt. Wären diese Informationen bekannt, könnten sie als weitere Metadaten in die Analyse einfließen. Wäre etwa bekannt, dass zwei oder mehrere IP-Adressen zum gleichen Bereich gehören, könnten TCP-Sessions zu unterschiedlichen IP-Adressen (und ggf. Domainnamen) dennoch in einen semantischen Zusammenhang gebracht werden.

Um den geographischen Bezugspunkt einer IP-Adresse zu eruieren, kann auf öffentliche Datenbanken wie GeoIP[15] oder IP2Location[16] zurückgegriffen werden. Um festzustellen, ob IP-Adressen ggf. dem gleichen Subnetz angehören, kann anhand der Netzmaske, die diese Datenbanken enthalten, die erste IP-Adresse eines Bereichs festgestellt werden. Gleicht sich die erste Adresse bei mehreren IP-Adressen im Packet Dump besteht eine semantische Relation.

3.3.4. Zusammenfassung der Analyse-Daten

Nach Aufbereitung des Packet Dumps gemäß den beschriebenen Schritten, liegen folgende Daten pro TCP-Session für die nachfolgende Klassifikation vor:

- IP-Adresse und Port des Zielservers.
- Geolocation-Informationen zu Land, Stadt und geogr. Koordinaten der IP-Adresse.
- Die erste IP-Adresse des Subnetzes, dem die IP-Adresse des Zielservers angehört.
- Der tatsächlich aufgerufene Domain-Name oder andernfalls der „Reverse Hostname“.
- Die relative Start- und Endzeit bzw. daraus folgend die Dauer der Sitzung.
- Die Anzahl der hoch- und runtergeladenen geladenen Bytes.
- Histogramme mit der Verteilung der übertragenen Bytes für Up- und Download.

Um die Klassifikation bzw. das semantische Netz zunächst zu trainieren, werden außerdem die Assoziationen zwischen Sessions und einzelnen Applikationen benötigt. Um diese Trainingsinformationen zu erhalten, kann auf NetGuard zurückgegriffen werden. Alle anderen, obig auflisteten Daten, können aus beliebigen Packet Dumps extrahiert und generiert werden.

3.4. Klassifikation

Nach Aufbereitung der TCP-Sessions und der erhobenen Attribute kann ein semantisches Netz trainiert werden, das die Abhängigkeiten untereinander modelliert. Der gewählte Ansatz der „Semantic Patterns Transformation“ [6] basiert dabei auf der Annahme, dass alle Daten entweder symbolisch oder numerisch dargestellt werden. Um auch symbolische Daten in einer Vektorrepräsentation darzustellen, werden die gegebenen Informationen über den kMeans-Algorithmus in Cluster unterteilt und treten so in eine relative Distanz zueinander. Diese wiederum kann als Vektor dargestellt werden und eignet sich auch für traditionelle Klassifikationsalgorithmen des „Supervised Learning“ wie Naive Bayes oder Support Vector Machines.

Für die in diesem Projekt verfolgten Analyse wurde zunächst ein Fokus auf die Gewinnung und Generierung von Daten gelegt, mit den ein semantisches Netz trainiert werden konnte. Die Wahl des optimalen Klassifikationsalgorithmus oder dessen „Precision“ für den jeweiligen Datensatz standen nicht im Vordergrund; könnten in einem Folgeprojekt aber detaillierter untersucht werden.

[15] https://www.maxmind.com/en/geoip2-services-and-databases
[16] http://www.ip2location.com

4. Empirische Evaluierung des Analyse-Ansatzes

Um die Tauglichkeit des Analyse-Konzepts in der Praxis zu erproben und auch im Hinblick auf die eingangs angeführten Zielsetzungen dieses Projekts, wurde das Konzept prototypisch implementiert und auf einen mitgeschnittenen Datensatz angewendet. Im Folgenden werden die vorgenommenen Experimente detaillierter erklärt und die jeweiligen Ergebnisse vorgestellt.

Das Ziel ist in jedem Fall, zu erkennen / klassifizieren, welche Anwendungen zu einem gewissen Zeitpunkt aktiv, für gewissen Datenaustausch (Byte-Verteilung im Histogramm) verantwortlich waren und zu identifizieren wie akkurat die Klassifikationsergebnisse mit einem einfachen Classifier sind. Weitere Untersuchungen könnten selbstverständlich auch andere Algorithmen berücksichtigen.

4.1. Untersuchter Datensatz

In der Annahme, dass Applikationen, die der gleichen Kategorie angehören oder vom gleichen Hersteller sind, ein ähnliches Verhalten aufweisen könnten, wurde für die hier vorgenommene Evaluierung der Fokus auf zwei Typen von Mobilanwendungen gelegt:

1. Anwendungen verschiedener Hersteller, die Messenger-Funktionalität bereitstellen. Durch die Zugehörigkeit zur gleichen Kategorie könnte das Verhalten dieser Anwendungen ähnlich sein. Die Intention ist somit, Charakteristika in den Daten zu finden, die diese Anwendungen voneinander unterscheiden.

2. Anwendungen des gleichen Herstellers, die für ähnliche Aufgaben designiert sind oder wo anzunehmen ist, dass sie herstellerbedingt ähnliche Domains aufrufen. Im konkreten Fall wird der Fokus dabei auf Applikationen von Google gesetzt, da dieser Hersteller Dienste sowohl über Cloud-Dienste mit variablen IP-Adressen anbietet als auch Ressourcen unter gemeinsam verwendeten Domains, wie etwa www.googleapis.com bereitstellt.

4.2. Klassifikation nach Mobilanwendung

Bei dieser Evaluierung wurde der vorliegende Datensatz zu gleichen Teilen in ein Trainings- und ein Testset aufgespalten. Ersteres besteht aus den im vorigen Abschnitt angeführten Daten inklusive der Assoziationen mit Anwendungen (im Folgenden „Labels“ genannt). Beim Testset wurden die Labels entfernt. Das Set ist insofern auch vergleichbar mit Traffic, der aus beliebiger Quelle, d.h. nicht NetGuard, stammt.

Bei dieser Analyse sollte herausgefunden werden, ob die zuvor erhobenen Daten eine prinzipiell hinreichend gute Klassifikation ermöglichen. Zu diesem Zweck wurde zunächst mit den für die TCP-Sessions jeweils vorliegenden Daten ein semantisches Netz trainiert. Durch Clustering mit kMeans wurden daraufhin symbolische Werte des Netzes zu Vektoren transformiert, numerische mit den Extrema der gegebenen Wertebereiche hin normalisiert.

Nach dem Trainieren des Netzes wurde das Testset (ohne Labels) auf das Netzwerk angewandt. Durch das Propagieren des Testset durch das Netzwerk wurde allen semantischen Patterns (mit den TCP-Sessions) des Testset Aktivierungswerte zugewiesen. Für jedes semantische Pattern einer TCP-Session extrahieren wir daher die Aktivierungswerte der Labels. Der höchste Wert repräsentiert jenes Label mit dem die TCP-Session im Netzwerk am stärksten aktivieren aktiviert wurde.

Wie in Abbildung 4 ersichtlich, wurde der Traffic von 4 unterschiedlichen Messenger-Anwendungen mit Trainings- und Testdaten klassifiziert. Die Identifikation erfolgt anhand ihrer Paketnamen:
Signal[17]: org.thoughtcrime.securesms, Telegram[18]: org.telegram.messenger,
WhatsApp[19]: com.whatsapp, Facebook Messenger[20]: com.facebook.orca

[17] https://play.google.com/store/apps/details?id=org.thoughtcrime.securesms
[18] https://play.google.com/store/apps/details?id=org.telegram.messenger
[19] https://play.google.com/store/apps/details?id=com.whatsapp
[20] https://play.google.com/store/apps/details?id=com.facebook.orca

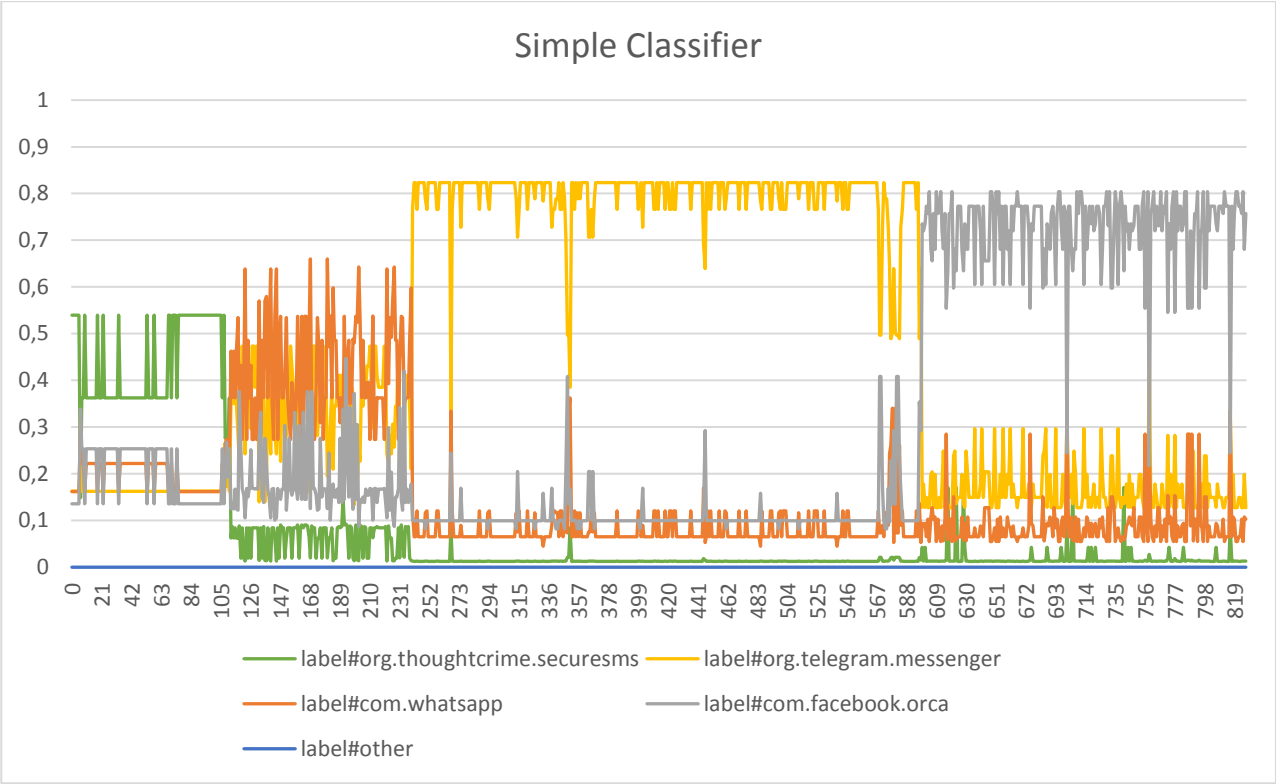

Abbildung 4. Einfache Klassifikation von Messenger-Anwendungen.

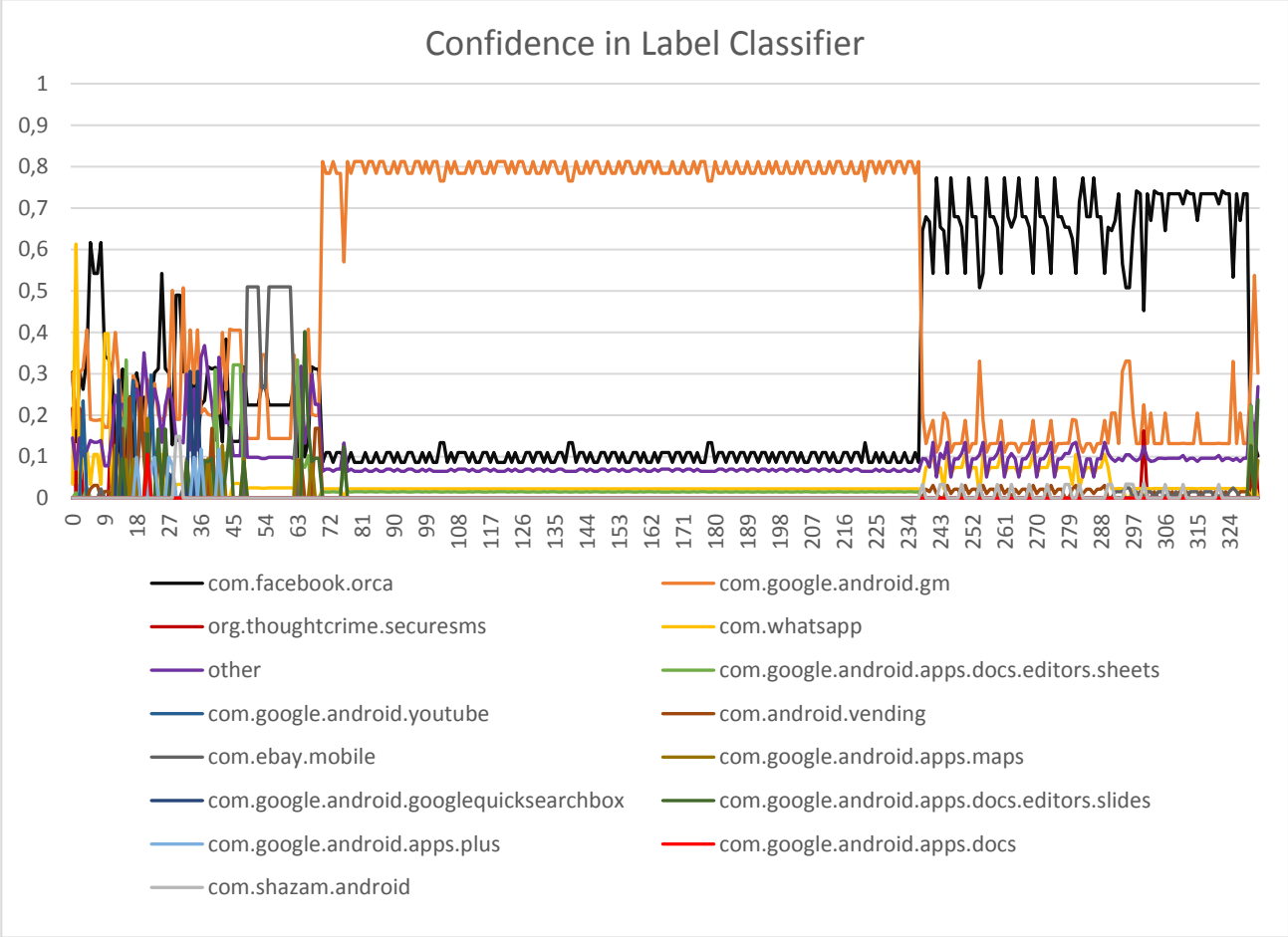

Abbildung 5. Aktivierungswerte der Labels bei einzelnen Sessions.

Auf der X-Achse sind die untersuchten TCP-Sessions entsprechend ihrer zeitlichen Abfolge angeordnet. Wie aus der Abbildung hervorgeht, waren insbesondere zu Beginn der Aufzeichnung mehrere Anwendungen gleichzeitig aktiv. Zu dem Zeitpunkt wo schließlich Sessions von WhatsApp dazu kommen, sinkt zugleich die Erkennung von Signal drastisch. Die Ursache dieses Verhaltens geht aus der Abbildung nicht hervor. Da WhatsApp und Signal jedoch (soweit bekannt) das gleiche Verfahren zur Verschlüsselung von Nachrichten implementieren, könnte dies mitunter die Änderung der Klassifikation dieser Sessions erklären. In etwa ab Session 123 ist aus er Grafik erkennbar, dass viele Sessions Telegram zugewiesen werden. Ungefähr ab Session 588 ist schließlich der Facebook Messenger vergleichbar aktiv.

Die empirische Erklärung für dieses Verhalten liegt in der Art und Weise wie die herangezogenen Analysedaten mitgeschnitten wurden. In etwa bis Session 123 wurde keine der Anwendungen aktiv verwendet, lief jedoch im Hintergrund des Betriebssystems (Austausch von Statusnachrichten, Synchronisierung der Kontakte, etc.), später wurden jedoch zunächst Telegram und dann Facebook Messenger aktiv verwendet. Praktisch ist daraus abzuleiten, dass die Klassifikation offensichtlich umso eindeutiger ist, wenn ein Messenger im Vordergrund verwendet wird.

Ein Vergleich des höchsten Aktivierungswert eines jeden Patterns mit den Informationen, die zuvor beim Extrahieren der Pattern-Features erstellt wurden, zeigt, dass die Klassifikation des Testset eine Genauigkeit von 83,3% aufweist.

4.3. Plausible Labels pro TCP-Session

Dieser (und alle folgenden) Versuche bauen auf den Erkenntnissen der einfachen Klassifikation von Messenger-Applikationen auf. Nachdem die vorherige Klassifikation keine Aussage darüber getroffen hat, welche Labels auf eine TCP-Session abgesehen von der klassifizierten noch zugetroffen hätten, sollte ermittelt werden, wie „eindeutig“ die Aktivierungswerte der Labels im Vergleich mit den sonstigen noch in Frage kommenden Labels für jede Session waren. Hierzu wurden für jede TCP-Session die fünf plausibelsten Labels ermittelt und die jeweiligen Aktivierungswerte in Relation zueinander gestellt.

Die Trainings- und Testdaten des vorherigen Schrittes umfassten lediglich TCP-Sessions, die zu Messenger gehörten. Für diese Analyse wurden nun auch weitere Anwendungen miteinbezogen, die in irgendeiner Art und Weise verwandt sind mit dem Messenger (siehe Abschnitt 4.1). Wie in Abbildung 5 erkennbar, korreliert die Konfidenz, dass Applikationen korrekt klassifiziert wurden vor allem dann, wenn Messenger aktiv verwendet wurden. Neben den „Top Labels“, die eindeutig für die Aktivierung der Patterns verantwortlich sind, zeigt die Grafik außerdem jene 4 Labels und deren Aktivierungswerte, die ein Pattern ebenfalls aktivieren. Unter dem Label „other“ wurden alle verbleibenden Labels zusammengefasst, die nicht in die „Top 5“-Labels fallen, um ein Pattern zu aktivieren. Die statistische Signifikanz dieser Label ist somit nicht wesentlich für die Klassifikation. Wie in der Abbildung ersichtlich, liegt eine Ungewissheit vor allem dann vor, wenn es sich um Traffic handelt, der im Hintergrund abläuft.

5. Fazit

Im Zuge dieses Projekts wurde ein Konzept erarbeitet, um den Traffic von Mobilanwendungen gezielt aufzuzeichnen, mit zusätzlichen Metainformationen die Informationsdichte aufzuwerten und auf seine Charakteristika hin zu klassifizieren. Der entworfene Ansatz eignet sich dafür, um Datenverkehr einzelner Anwendungen aus einem kollektiven Packet Dump zu extrahieren. Mit einem Fokus auf TCP-Sessions gelingt es, eine Abstraktionsebene einzuführen, dank der eine Analyse nicht mehr auf Ebene einzelner Pakete erfolgen muss sondern zudem mit verschlüsselten, gleich wie unverschlüsselten Daten umgehen kann. Die Heterogenität (symbolische und numerische Werte) der so erhobenen Netzwerkdaten erfordert jedoch weitere Verarbeitungsmaßnahmen um die Relationen einzelner Werte semantisch korrekt abzubilden. Nach einer Studie mehrerer Ansätze wurde für dieses Projekt das Konzept der „Semantic Patterns Transformation“ eingesetzt. Wie sich herausgestellt hat, ist diese Technik für eine Klassifikation der Anwendungen anhand ihres Datenverkehrs gut geeignet.

Die eingangs in diesem Bericht angeführten Fragestellungen wurden im Laufe des Projekts nicht in jedem Aspekt vollständig beantwortet. Ungeachtet dessen wurde in diesem Projekt ein fundamentales Verständnis über das Verhalten mobiler Anwendungen aus ihrem Datenverkehr abgeleitet. Die aus diesem Projekt gewonnenen Erkenntnisse dienen einerseits als Wissensbasis für ähnliche Untersuchungen im Netzwerkbereich und liefern andererseits aufschlussreiche Erkenntnisse, die wiederum positiv bei der Inspektion beliebiger Mobilanwendungen beitragen können.

Eine wesentliche Erkenntnis aus der Analyse ist, dass sich Applikationen anhand ihres Datenverkehrs tatsächlich individuell identifizieren lassen. Da immer mehr Datenverkehr verschlüsselt stattfindet, war es notwendig, eine Basis zu finden, auf der die Analyse ungeachtet von Verschlüsselung funktionieren kann.

Die Frage, welche Attribute innerhalb eines „Semantic Patterns“ eine TCP-Session möglichst genau charakterisiert, konnte in diesem Projekt ressourcenbedingt nicht berücksichtigt werden. Während dem Projekt sind auch weitere Fragestellungen aufgetaucht, deren Beantwortung im Kontext der Thematik relevant sein könnte: Bei der Rekonstruktion von Domainnamen durch DNS-Matching spielte die zeitliche Abfolge von TCP-Sessions eine wesentliche Rolle. Diese zeitliche Komponente könnte auch in die Analyse einfließen, indem analysiert wird ob Sessions eine zeitliche Abhängigkeit voneinander haben. Dies wäre z.B. der Fall, wenn TCP-Sessions immer zunächst zu einem und später zu einem anderen Host aufgebaut werden würden.

Grundsätzlich ließe sich die Klassifikation einzelner Applikationen auch abstrahieren werden zu „Gruppen“. Es stellt sich etwa die Frage, ob Anwendungen, die der gleichen Kategorie angehören, sich im Vergleich mit anderen Kategorien auch im Traffic unterscheiden. Die Abstraktionsebene würde somit von einzelnen Anwendungen zu Kategorien steigen. Dies könnte wiederum dazu eingesetzt werden um auch unbekannten Anwendungen ausgehend von ihrem Traffic eine Kategorie zuzuweisen.

6. Literaturverzeichnis

[1] S. Valenti, D. Rossi und A. Dainotti, „Reviewing Traffic Classification,“ *Data Traffic Monitoring and Analysis,* 2013.

[2] T. Nguyen und G. Armitage, „A Survey of Techniques for Internet Traffic Classification Using Machine Learning,“ *IEEE Communications Surveys & Tutorials, Vol. 10, No. 4,* 2008.

[3] E. Biersack, C. Gallegari und M. Matijasevic, Data Traffic Monitoring and Analysis, Springer, 2013.

[4] A. Moore und K. Papagiannaki, „Toward the accurate identification of network applications,“ *Proc. Passive and Active Measurement Workshop (PAM 2005),* 2005.

[5] S. Deerwester, „Indexing by Latent Semantic Analysis,“ *Journal of the American society for information science,* 1990.

[6] P. Teufl, H. Leitold und R. Posch, „Semantic Pattern Transformation: Applying Knowledge Discovery Processes in Heterogeneous Domains,“ *Proceedings of the 13th International Conference on Knowledge Management and Knowledge Technologies,* 2013.