Fraunhofer
SIT

FRAUNHOFER-INSTITUT FÜR SICHERE INFORMATIONSTECHNOLOGIE

APPICAPTOR SECURITY INDEX 9/2018

Appicaptor Security Index
9/2018

Der Einsatz von Apps in Unternehmen erfordert einen kritischen Blick auf die Risiken, um durch Prüf- und Freigabemechanismen einer Gefährdung effektiv begegnen zu können. Im Folgenden werden Ergebnisauszüge automatisierter Appicaptor-Analysen für die Top 2.000 der kostenlosen iOS- und Android-Apps vorgestellt.

Prüfung der App-Märkte

Die etablierten App-Märkte für iOS und Android bieten eine Fülle nützlicher Apps. Dabei soll die Prüfung der App-Markt-Betreiber und die IT-Sicherheitskonzepte der Smartphones für eine sorgenfreie Nutzung sorgen. Und in der Tat können diese App-Märkte ihren Nutzern ein deutliches Sicherheitsplus im Vergleich zu ungeprüften alternativen Softwarequellen bieten, wenn es darum geht, größere Malware-Angriffswellen abzuwehren. So schätzt Google die Zahl der Geräte mit möglicherweise schädlichen Apps auf 0,05% bei ausschließlicher Nutzung des Google Play Stores gegenüber 0,71% bei der Nutzung alternativer Quellen[1].

Die Sicherheitsqualität von Apps bleibt hingegen für die Nutzer nicht ersichtlich. Nutzer, die Apps mit einer schlechten Sicherheitsqualität verwenden, können über Schwachstellen seriöser Apps daher dennoch Opfer von Angriffen werden, ohne dass sich Malware auf ihrem Smartphone befindet.

Für IT-Verantwortliche stellt sich daher die Frage, welche Risiken durch die Verwendung einer App für ihr Unternehmen bestehen und ob diese tragbar sind oder nicht. Erst mit diesen Informationen über Apps besteht die Möglichkeit, durch App-Freigabekonzepte Risiken bei der geschäftlichen Smartphone-Nutzung für Unternehmen zu erfassen und hinsichtlich der Sicherheitsanforderungen der Einsatzumgebung sinnvoll zu begrenzen.

Schwachpunkt: Kommunikation

Nach wie vor ist einer der häufigsten Schwachpunkte bei Apps eine ungenügende Absicherung der Kommunikation (siehe Kasten: Kommunikationsrisiken). Dadurch wird die Nutzung öffentlicher WLAN-Zugänge zum direkten Angriffspunkt auf Unternehmensdaten, wenn diese mit verwundbaren Apps bearbeitet werden. Angreifer mit umfangreicheren Mitteln können ungeschützte Kommunikationsverbindungen aber auch in Mobilfunknetzen und Internetroutern einsehen und manipulieren[2].

Apps mit fehlender oder unsicherer Kommunikationsabsicherung ermöglichen Angreifern, die schlechte Sicherheitsqualität auszunutzen und Zugriff auf übertragene Unternehmensdaten oder Passwörter der Mitarbeiter zu erlangen.

In den aktuellen Analysen der Top 2.000 kostenloser iOS-Apps mit Appicaptor (siehe Kasten Appicaptor: Analyse der Sicherheitsqualität) zeigt sich, dass immer noch 60% der Apps HTTP-Verbindungen nutzen, um Inhalte wie HTML-Seiten und JavaScript-Code zu laden (Android: 75%). Das betrifft aber nicht nur News- oder Taschenlampen-Apps, sondern beispielsweise auch 71% der wesentlich kritischeren File-Viewer-Apps (Android: 82%). Im Vergleich zu 2016 ist damit bei Android der Anteil unverschlüsselter Kommunikation nur um etwa 11% gesunken, während bei iOS etwa 22% Rückgang zu verzeichnen sind.

Dies ist vermutlich auch der Initiative von Apple zuzuschreiben, dem Risiko mit sicheren Standardeinstellungen für die Kommunikationsabsicherung entgegenzuwirken. Mit der in iOS 9 eingeführten App Transport Security (ATS) wird aktuell für 98,7% der Apps (siehe Abbildung 10), die auf dem Softwareentwicklungsbaukasten (SDK) für iOS 9 und höher aufbauen, die unverschlüsselte Kommunikation über HTTP unterbunden und die TLS 1.2 Protokollversion mit dem derzeit besten Sicherheitsniveau erzwungen. Erst durch das Erstellen von Ausnahmeregeln können unsicherere Einstellungen für Apps aktiviert werden. Die für Anfang 2017 angekündigte[3] schärfere Prüfung dieser Ausnahmen hat im Vergleich zu 2016 allerdings noch nicht dazu geführt, dass

[1] »Android Security 2017 Year in Review«, März 2017
[2] https://www.xda-developers.com/4g-lte-vulnerability-enables-eavesdropping-on-conversations-and-all-data-traffic/
[3] WWDC 2016: https://developer.apple.com/videos/playwwdc2016/706/?time=243

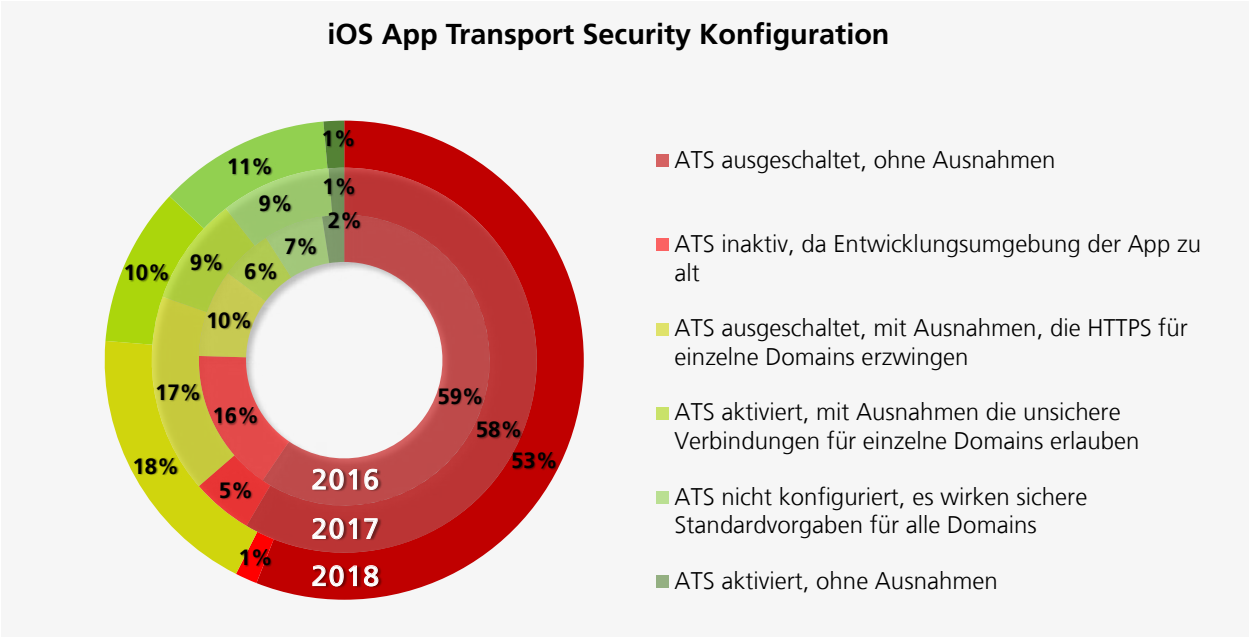


Abbildung 1: Aufteilung der Verwendung von App Transport Security bei den kostenlosen Top 2.000 iOS-Apps (Appicaptor, September 2018)

sich der Anteil der Apps, die ATS vollständig abschalten, wesentlich reduziert hat. Dieser Anteil ist mit aktuell 53% im Vergleich zu den Vorjahren mit etwa 58% kaum gesunken (siehe Abbildung 1). So hat auch Apple in der WWDC 2017 festgestellt, dass die Nutzung von ATS mit Ausnahmen weiter zugenommen hat, sodass Apple die Entwickler und Server-Betreiber weiter ermutigte, ATS vollständig zu aktivieren bzw. zu unterstützen[4]. Der gestiegene Anteil der Apps mit ATS-Ausnahmen erfolgt offenbar nur zugunsten von Apps, die zuvor ATS aufgrund der zu alten Entwicklungsumgebung gar nicht nutzen konnten. Dieser Anteil ist deutlich von 16% in 2016 auf 1% in 2018 gefallen und zeigt die erfolgreiche Aktualisierung von Apps im iOS-Appstore. Trotz der inzwischen günstigeren Möglichkeiten für TLS-Zertifikate (z.B. letsencrypt.org) und der Initiative von Apple und Google[5] scheint dennoch für viele Entwickler immer noch die Notwendigkeit für HTTP-Ausnahmen zu bestehen, wenn eingebundene Dienstleister für Werbung, Tracking und andere Angebote keine sichere Kommunikation anbieten.

[4] WWDC 2017: https://developer.apple.com/videos/play/wwdc2017/701/?time=1865

[5] https://webmasters.googleblog.com/2014/08/https-as-ranking-signal.html

HTTPS unsicher umgesetzt

Aber auch bei Apps, die HTTPS verwenden, ergeben sich immer wieder Schwachstellen bei der Prüfung der Server-Zertifikate. Die Prüfung übernimmt normalerweise das Betriebssystem, sie kann aber auch vom Entwickler verändert werden. Dies wird während der Entwicklungszeit beispielsweise vorgenommen, um zu Servern TLS-Verbindungen aufzubauen, die noch über kein passendes Zertifikat verfügen.

Wie in Abbildung 2 beschrieben, kann beispielsweise unter iOS die Zertifikatsprüfung durch Aufruf von `setValidateSecureCertificate` mit dem Parameter `false` für eine Verbindung der App abgeschaltet werden. Werden dann in der Entwicklungszeit gültige Serverzertifikate installiert, kann die Abschaltung dennoch unbemerkt bleiben. Solche Altlasten aus der Entwicklungszeit ermöglichen es Angreifern, sehr einfach auf die kommunizierten Inhalte zuzugreifen. Im dargestellten Beispiel können bei der Send Word Now for iPhone App (Version 2.9.4) beispielsweise Benutzername und Passwort berechtigter Nutzer für die Notfall- und Massenkommunikationsplattform der Firma OnSolve abgefangen werden.

Andere Apps implementieren eigene Zertifikatsprüfungen durch sogenanntes Certificate Pinning, um zu verhindern, dass gefälschte Zertifikate von unterwanderten Certification Authorities genutzt werden können. Dieser für Apps durchaus sinnvolle Zusatzschutz enthält in der Praxis allerdings eine Vielzahl von zu beachtenden Fallstricken. Andernfalls verwandelt sich der Zusatzschutz in eine massive Verwund-

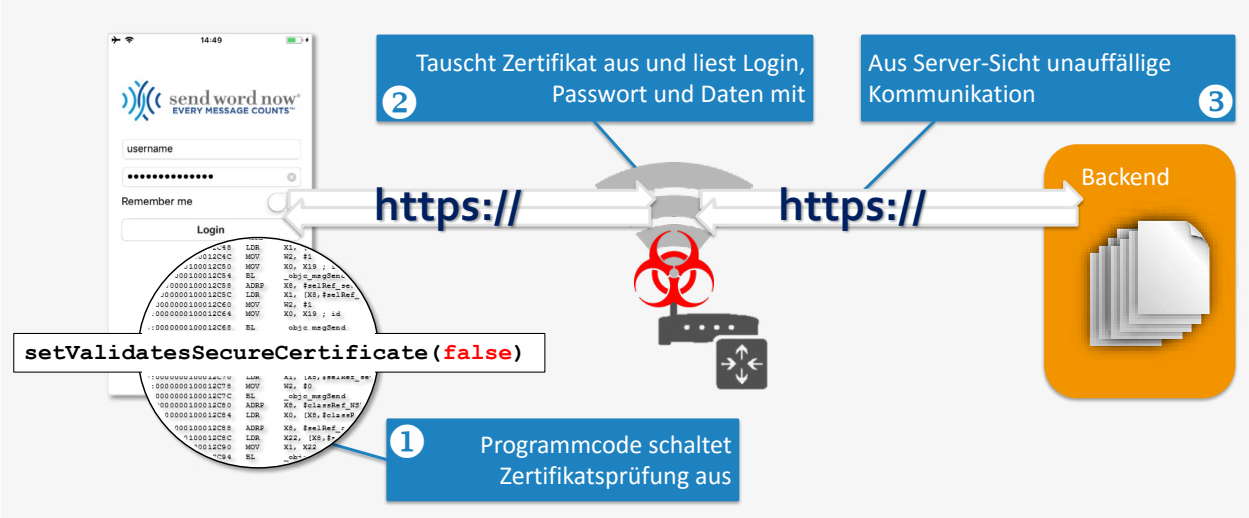


Abbildung 2: Altlasten aus der Entwicklungszeit: Gefährdung der Kommunikationsicherheit durch Abschalten der Server-Zertikatsprüfung, die in der Produktivversion immer noch aktiv ist. Hier dargestellt mit der Send Word Now for iPhone App (Version 2.9.4)

barkeit, sodass dies besser durch eine anerkannte bestehende Bibliothek erfolgen sollte als durch eine eigene Umsetzung[6].

Insgesamt enthielten 22% der getesteten iOS-Apps eine modifizierte Zertifikatsverifikation und 5,7% unsicheren Programmcode für die Zertifikatsverifikation.

Hybrid-Apps: Web-Risiken in mobilen Apps

Noch gravierender werden Kommunikationsrisiken, wenn die App-Funktionalität durch Webtechnologien gesteuert wird. Diese Technik, bei der App-Logik und Darstellung mittels HTML und JavaScript realisiert werden, wird immer beliebter. Sie verspricht Kostenreduktion durch plattformunabhängige Programmierung, da in der Theorie die Darstellung über HTML und die Steuerung der Abläufe mittels JavaScript nur einmal erzeugt werden muss. Den Zugriff auf Smartphone-Ressourcen übernehmen dann plattformspezifische Programmmodule, die in nativen Bibliotheken, wie etwa Apache Cordova, für viele Smartphone-Plattformen bereits kostenlos verfügbar sind.

Die so entstehenden Hybrid-Apps müssen aber nicht nur in der Praxis dann doch oft noch für spezifische Plattformeigenheiten angepasst werden, sie haben auch einen gravierenden IT-Sicherheitsnachteil: Die gesamte Programmlogik wird in diesen Apps durch unsignierten JavaScript-Programmcode gesteuert. Dies kann durch Angreifer auf verschiedene Weise missbraucht werden. Werden beispielsweise Inhalte ungeschützt nachgeladen, so kann ein Angreifer den fehlenden Integritätsschutz ausnutzen und die Inhalte manipulieren. So lassen sich oft direkt JavaScript-Dateien manipulieren und dadurch auch die Programmlogik ändern.

Abbildung 3 zeigt dies am Beispiel einer vertrauenswürdigen Nachrichten-App. Aufgrund einer Injection-Lücke, einer für Web-Anwendungen typischen Schwachstelle durch fehlende Integritäts- und Inhaltsprüfung, können Angreifer den JavaScript-Code zur Laufzeit durch einen Man-in-the-Middle-Angriff ändern. Die bestehende Berechtigung für den Zugriff auf Mikrofon und Dateisystem konnte dann missbraucht werden, um unbemerkt Audio-Mitschnitte der Umgebung von Nutzern der "rbb24-App" (und anderen Cordova-Apps mit ähnlichen Schwächen) anzufertigen und Inhalte der SD-Karte unbemerkt auszulesen. Audio-Mitschnitte und Inhalte der SD-Karte bei Android, wie etwa PDFs aus Download- oder Attachment-Ordnern, können dann unbemerkt in einen Cloudspeicher des Angreifers kopiert werden. Dies funktioniert bei Android auch dann noch, wenn die App bereits gewechselt oder der Bildschirm abgeschaltet wurde.

Ähnliche Verwundbarkeiten von Hybrid-Apps betreffen aber auch andere Ressourcen wie Telefonbuch, Kalender, Zwischenablage, Positionsdaten bis hin zum Zugriff auf lokale

[6] https://www.owasp.org/index.php/Certificate_and_Public_Key_Pinning

Abbildung 3: Funktionsmissbrauch: Fehlender Integritätsschutz stellt eine große Gefahr für Hybrid-Apps dar, wenn die App Berechtigungen für den Zugriff auf kritische Ressourcen wie Mikrofon oder Dateisystem haben. Hier dargestellt am Fall der rbb|24 App für iOS und Android (Version 1.7.1.)

Daten in der Sandbox der verwundbaren App. In der Menge der analysierten Cordova-iOS-Apps weisen aktuell 71,4% (Android 73,6%) die Voraussetzung für eine Manipulation der App über Web-Inhalte auf, um die JavaScript-Brücke zu den Smartphone-Ressourcen anzugreifen (siehe Abbildung 4). Die Auswirkungen unterscheiden sich dabei allerdings stark, je nach verwendeten Smartphone-Ressourcen und verarbeiteten Daten. Durch die Möglichkeit, diese Faktoren automatisiert in einer Code-Analyse zu berücksichtigen, kann jedoch eine genauere Risikobewertung erfolgen, sodass nur solche Apps von der Nutzung im Unternehmen ausgeschlossen werden, von denen ein konkretes Risiko ausgeht.

Weitere Risiken für Cordova Apps bestehen aber auch in der Nutzung bekannter verwundbarer Cordova-Versionen, auf denen 10,2% bei iOS und 5,7% der Android-Apps basieren. Dies ermöglicht es Angreifern, bspw. unter Android bei Cordova-Versionen vor 3.5.1[7], per Link die Cordova-App mit anderem HTML-Seiteninhalt zu starten, um dadurch die verfügbaren Plugins mit Zugriff auf kritische Ressourcen für ihre Zwecke zu missbrauchen.

Aber auch der Anteil bekannter verwundbarer JavaScript-Bibliotheken in Cordova-Apps stellt mit 28,6% (Android 28,7%) ein großes Problem dar. So ermöglicht bspw. die Verwendung der verwundbaren Angular.js Version 1.4.4 es Angreifern durch manipulierte Nutzereingaben aus eigentlich gesicherten `ng-bind-html` Umgebungen auszubrechen. Diese Umgebung erlaubt es normalerweise nur, HTML-Formatierungen zu verwenden, Aufrufe von JavaScript oder anderen aktiven Inhalten soll unterbunden werden. Die bekannte Schwäche dieser Version erlaubt es Angreifern, den Schutzmechanismus auszutricksen, wodurch wieder alle zur Verfügung stehenden Berechtigungen der App durch einen Angreifer missbraucht werden können.

Auf der anderen Seite stellt Cordova aber auch erweiterte Schutzmechanismen zur Verfügung. Die Whitelist-Konfiguration soll bspw. verhindern, dass Inhalte wie HTML-Seiten von nicht autorisierten Domains in den sicherheitskritischen WebView-Browser der Cordova-App geladen werden. Am sichersten wäre es beispielsweise, nur lokale Inhalte zuzulassen, sodass ein Angreifer keine Möglichkeit erhält, die Anwendungslogik durch externe Seiten zu verändern. Werden dennoch externe Inhalte benötigt, kann durch die Whitelist der Zugriff auf die berechtigten Domains eingeschränkt werden. Leider enthält jedoch der Standard-Whitelist-Eintrag ein Stern-Zeichen, was den Zugriff auf alle Domains erlaubt. Obwohl die Cordova-Dokumentation ausdrücklich darauf hinweist[8], diesen Eintrag vor Veröffentlichung der App auf die tatsächlich notwendigen Domains zu ändern, nutzen 93,9% der Cordova-iOS-Apps (Android 94,3%) den Standard und damit die schwächste mögliche

[7] CVE 2014-3500: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-3500

[8] https://cordova.apache.org/docs/en/latest/guide/appdev/whitelist/index.html

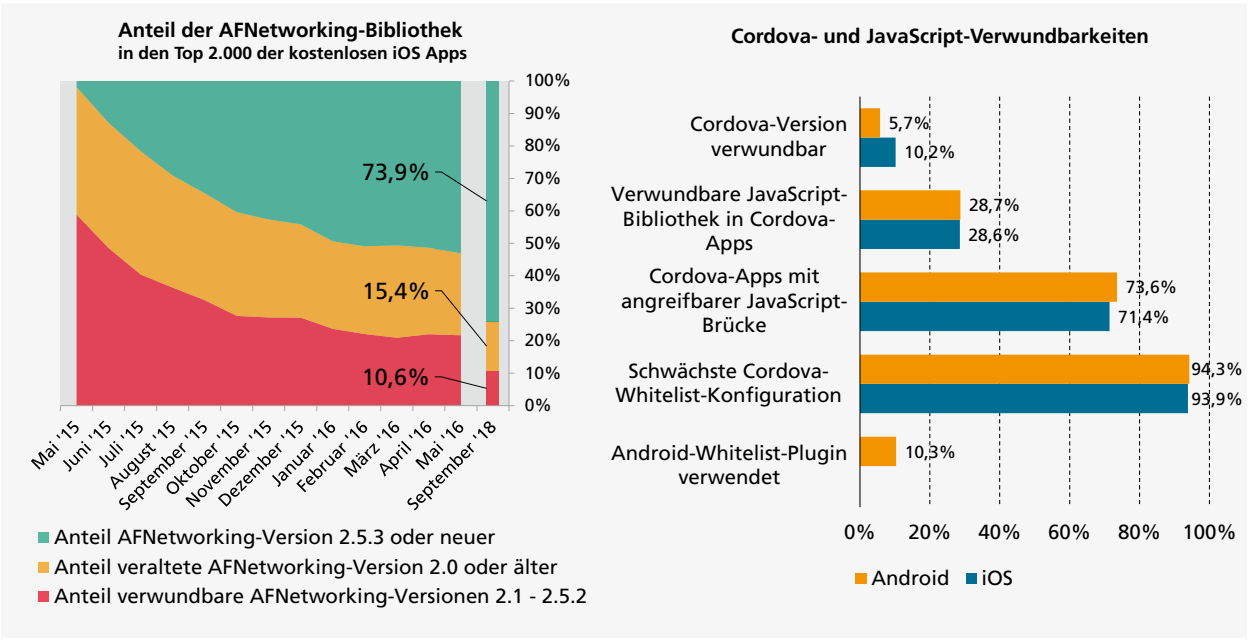


Abbildung 4: Auch 14 Monate nach der Veröffentlichung einer gravierenden Schwachstelle in der AFNetworking Bibliothek für iOS enthielten 20% der kostenlosen Top 2.000 verwundbare Versionen. Nach über 3 Jahren sind immer noch 10% der genutzten AFNetworking-Versionen verwundbar. Auch in Hybrid-Apps werden verwundbare Versionen noch verwendet. (Appicaptor, September 2018)

Cordova-Whitelist-Konfiguration. Ebenso wird der Rat, das Android-Whitelist-Plugin zu verwenden, nur bei 10,3% der Apps befolgt. Neben dem großen Risikopotenzial durch bereits bekannte Schwachstellen zeigen diese Zahlen somit auch: Entwickler nutzen kaum die verfügbaren Schutzmechanismen und machen damit Hybrid-Apps anfälliger als sie sein müssten. Denn noch striktere Sicherheitsmaßnahmen wie die Verwendung von Content Security Policies (CSP)[9], die beispielsweise auch Inline-JavaScript unterbinden könnten, konnten in bisherigen Analysen nur sehr vereinzelt angetroffen werden. Neben der Nutzung von HTTPS und der kontinuierlichen Aktualisierung von Fremdbibliotheken kann mit einer strikten CSP ein Großteil des Risikos von Hybrid-Apps reduziert werden.

Allerdings besteht für Hybrid-Apps nicht nur die Gefahr, dass Schwächen in der Web-Sicherheit durch Angreifer ausgenutzt werden. Auch der App-Hersteller selbst kann die Funktionsweise der App jederzeit ändern, ohne erneut die App durch den Review-Prozess der App-Märkte prüfen zu lassen. Zudem kann bei Hybrid-Apps, die Inhalte und Programmcode von Drittanbietern integrieren, auch eine Manipulation von deren Seite erfolgen. Beispielsweise wenn die App für die Darstellung von Werbung oder zur Analyse von App- und Nutzerverhalten JavaScript-Code von externen Quellen nachlädt. So enthalten bereits 45% mindestens eine benannte JavaScript-Brücke, über die Funktionen im nativen App-Teil aufgerufen werden können.

Schlechte oder fehlende Kryptografie

Ein wesentlicher Faktor der Sicherheitsqualität von Apps basiert auf der verwendeten Kryptografie, da diese für viele Sicherheitsfunktionen von Apps notwendig ist. Häufigster Einsatzzweck ist der Schutz der Vertraulichkeit von Daten und die Prüfung der Authentizität von Kommunikationspartnern. Dabei kommt es zum einen darauf an, dass Kryptografie richtig verwendet wird und zum anderen, dass keine veralteten Verfahren oder zu kurze Schlüssellängen eingesetzt werden, die inzwischen als unsicher oder gar gebrochen gelten[10].

Für den Fall der unsicheren Verwendung zeigt Abbildung 5 anhand der MultipeerConnectivity API von iOS ein Beispiel für den Einfluss auf die Sicherheitsqualität der App. Diese API erlaubt es Entwicklern, sehr einfach einen direkten Austausch von Daten zwischen Endgeräten über drahtlose Kommunikation zu realisieren. Vorgesehen ist dabei bereits, dass dies sowohl authentisiert als auch verschlüsselt passiert,

[9] https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP

[10] Technische Richtlinie BSI TR-02102-1 https://www.bsibund.de/SharedDocs/Downloads/DE/BSI/Publikationen/TechnischeRichtlinien/TR02102/BSI-TR-02102.html

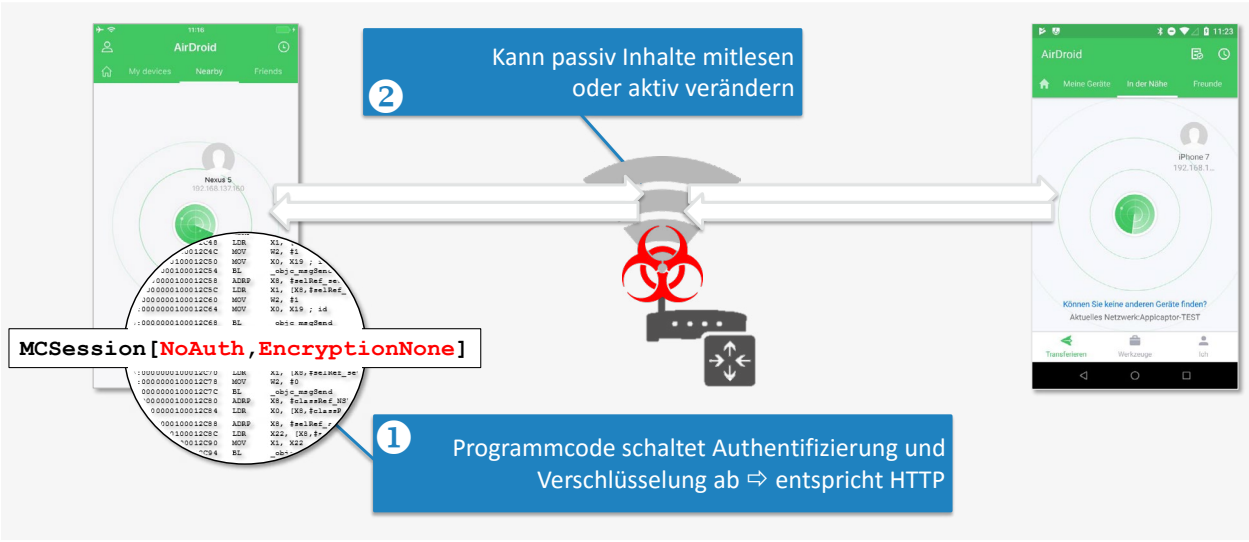


Abbildung 5: Schlechte / Fehlende Kryptographie: Gefährdung von Unternehmensdaten bei der Peer-To-Peer-Übertragung durch fehlende Verschlüsselung und Authentizierung. Hier dargestellt an der AirDroid für iOS (Version 1.0.3)

was durch den Entwickler jedoch entsprechend umgesetzt bzw. konfiguriert werden muss. Hier zeigen die Appicaptor-Analysen, dass von den iOS-Apps mit dieser Funktionalität 40% die Übertragung weder verschlüsseln noch die Kommunikationspartner authentisieren. Wie am Beispiel der AirDroid-iOS-App (Version 1.0.3) dargestellt, kann ein Angreifer dadurch passiv die Übertragungen in der Umgebung mitlesen. Bei 20% wird immerhin verschlüsselt übertragen, aber ohne die Authentizität des Kommunikationspartners zu prüfen. Hier wäre dann immer noch ein aktiver Man-in-the-Middle-Angriff möglich.

Risiken durch Fremdbibliotheken

Programmcode aus externen Quellen ist aber auch in nativen Apps sehr häufig anzutreffen. Diese Code-Bibliotheken bieten eine Fülle an Möglichkeiten, die nicht mehr selbst programmiert werden müssen, was die App-Entwicklungskosten reduzieren kann. So werden durch Entwickler häufig viele Bibliotheken eingebunden, aus denen aber oft nur eine kleine Untermenge der vorhandenen Funktionalität genutzt wird. Dabei erfolgt die Auswahl häufig nur nach Entwicklervorlieben und Gewohnheit als aus einer Notwendigkeit der Spezifikation.

Im Falle von Schwachstellen in externen Bibliotheken ergibt sich daraus das Problem, dass Entwickler häufig die Meldung neuer Verwundbarkeiten in den von ihnen verwendeten Bibliotheken übersehen, die damit aber auch die eigene App betreffen können. So zeigt das Beispiel einer veröffentlichten Verwundbarkeit der beliebten AFNetworking-Bibliothek für iOS, dass auch 14 Monate nach der Meldung einer kritischen Verwundbarkeit noch 20% der kostenlosen Top 2.000 iOS-Apps, die diese Bibliothek verwenden, eine verwundbare Version einsetzen (siehe Abbildung 4). Auch nach mehr als 3 Jahren sind immer noch 10% der genutzten AFNetworking-Versionen verwundbar. Diese Versionen erlauben es Angreifern, die mit der Bibliothek abgesicherte HTTPS-Kommunikation mitzulesen, da durch die Verwundbarkeit die Serverzertifikate der Kommunikationspartner nicht korrekt geprüft werden.

Bei Fehlern in Betriebssystem-Bibliotheken profitieren indes alle Apps automatisch von Betriebssystem-Updates, weshalb Entwickler versuchen sollten, zunächst deren Funktionalität voll auszuschöpfen, bevor externe Bibliotheken zum Einsatz kommen. Dieses Beispiel zeigt auch, dass einmal eingebundene Bibliotheken kaum aktualisiert werden: Der Anteil der inzwischen sehr veralteten AFNetworking-Versionen 2.0 und kleiner nimmt nur sehr langsam ab, und das, obwohl 98% der Apps seit Bekanntwerden der Schwachstelle mindestens einmal aktualisiert wurden.

Klassifizierung von App-Funktionen für Risikobewertung

Die beispielhaft dargestellten Risiken zeigen den Einfluss der Sicherheitsqualität von Apps auf das Angriffspotenzial. Bei der Bewertung des daraus resultierenden Risikos für Unternehmen kommt es aber immer darauf an, welche Funktion eine App erfüllt und mit welchen Daten sie arbeitet bzw. worauf sie potenziell Zugriff erlangen könnte.

Kommunikationsrisiken

Kommunikationsrisiken beziehen sich auf den fehlenden, schwachen oder fehlerhaften Schutz der Geheimhaltung und Integritätssicherung von Informationen während eines Informationsaustauschs mit externen Quellen. Gründe für die Einstufung einer App in dieser Kategorie sind beispielsweise folgende:

TLS-Schwachstelle: Die App beinhaltet unsicheren Code zum Schutz der Kommunikation mit SSL/TLS. Oftmals ist unsicherer Code die Ursache für fehlerhaften Schutz vor Man-in-the-Middle Angriffen. Die Umsetzung korrekter Secure Socket Layer (SSL) oder Transport Layer Security (TLS) Kommunikation kann in der App-Entwicklung prinzipiell einfach mit den Standardfunktionen des Smartphone-Betriebssystems durchgeführt werden. In der Entwicklungsphase einer Smartphone-App wird jedoch die SSL/TLS-Konfiguration oder ihre Prozesse häufig modifiziert, um das Debugging oder die Funktion in einer Testumgebung ohne gültige Zertifikate zu ermöglichen. Dies wird benötigt, wenn die Test-Umgebung oder -weitaus schlimmer- die Produktivumgebung kein Server-Zertifikat verwendet, das durch eine Certificate Authority unterzeichnet wurde. App-Entwickler lösen dieses Problem durch die Deaktivierung oder Änderung der SSL/TLS-Sicherheitsmaßnahmen.

Ungeschützte Kommunikation: Die Verwendung des ungeschützten HTTP-Protokolls zur Übertragung von Parametern oder zur Abfrage von Inhalten von Servern, welche eigentlich fähig wären eine geschützte HTTPS-Kommunikation aufzubauen. Oft wird von den Entwicklern argumentiert, dass der ungeschützte Zugriff über HTTP nicht problematisch sei, da die übertragenen Informationen nicht vertraulich wären. Dies berücksichtigt jedoch nicht, dass ein Angreifer jede ungeschützte Kommunikation nicht nur lesen sondern auch manipulieren kann. Dies gibt einem potentiellen Angreifer die Möglichkeit, Server-Anfragen oder -Antworten zu verändern (oder mit eigenen Funktionen zu ergänzen) und damit die auswertende App-Umgebung zu einem anderen Verhalten (im Bezug auf das Verhalten mit unmodifizierten Daten) zu bewegen. Dies kann u.a. verwendet werden, um das Vertrauen des Benutzers in eine App auszunutzen, bspw. durch eine hinzugefügte Dialogbox mit Passwortabfrage, deren Eingaben an den Angreifer gesendet werden.

Implementierungsfehler: Fehlender oder mangelhafter Schutz gegen Injection-Angriffe. Angreifer können dann Daten manipulieren, die durch den Implementierungsfehler als Programminstruktionen verstanden werden und das Verhalten der App ändern können.

Apps, die aufgrund fehlender Berechtigungen keinen Zugriff auf Smartphone-Ressourcen erlangen können, haben selbst bei gravierenden Schwächen ein geringeres Risikopotenzial, da die Sandbox-Konzepte der Smartphone-Betriebssysteme in diesem Fall meist eine Ausweitung des Zugriffs auf kritische Ressourcen unterbinden können. Dennoch ist auch in diesem Fall zu berücksichtigen, für was die App eingesetzt wird, um z.B. ein mögliches Abgreifen von Unternehmenspasswörtern in die Risikobetrachtung mit einzubeziehen. Daher reicht die Berücksichtigung der Berechtigungen allein nicht aus, es muss auch die übliche Verwendung der App mit einfließen.

Bei der automatischen Analyse müssen dazu Informationen über die generelle App-Funktion erfasst werden. Die Kategorisierung in den App-Märkten reicht dazu jedoch nicht aus, da beispielsweise in der App-Markt-Kategorie Finanzen sowohl die unkritischen Taschenrechner- und Börsenkurs-Apps zu finden sind als auch die wesentlich kritischeren Mobile-Banking-Apps.

Appicaptor begegnet diesem Problem durch eine Klassifizierung der Apps anhand ihrer Beschreibungstexte. Mittels eines Machine-Learning-Ansatzes werden Apps der gleichen Funktionsklasse erkannt, sodass aus der Funktionsklasse mittels ihres jeweils individuellen Risikomodells eine Bewertung der analysierten Schwachstellen erfolgen kann. Es erfolgt somit zum einen ein Abgleich der erwartbaren mit der vorgefundenen Funktionalität, zum anderen eine Ableitung der Auswirkungen von entdeckten Schwächen für die jeweilige Funktionsklasse. So sind detektierte Schwächen bei kryptografischen Verfahren für die Funktionsklasse der Passwortmanager schwerwiegender als für Taschenlampen-Apps. Desweiteren fließt in die Risikomodelle der Ressourcenzugriff auf Sensoren und Daten mit ein. Die Information, dass für eine App die Verarbeitung von Office-Dokumenten detektiert wurde, kann dann gegen die Funktionsklasse auf Plausibilität geprüft werden und erhöht somit die Auswirkungen bei detektierten Schwächen.

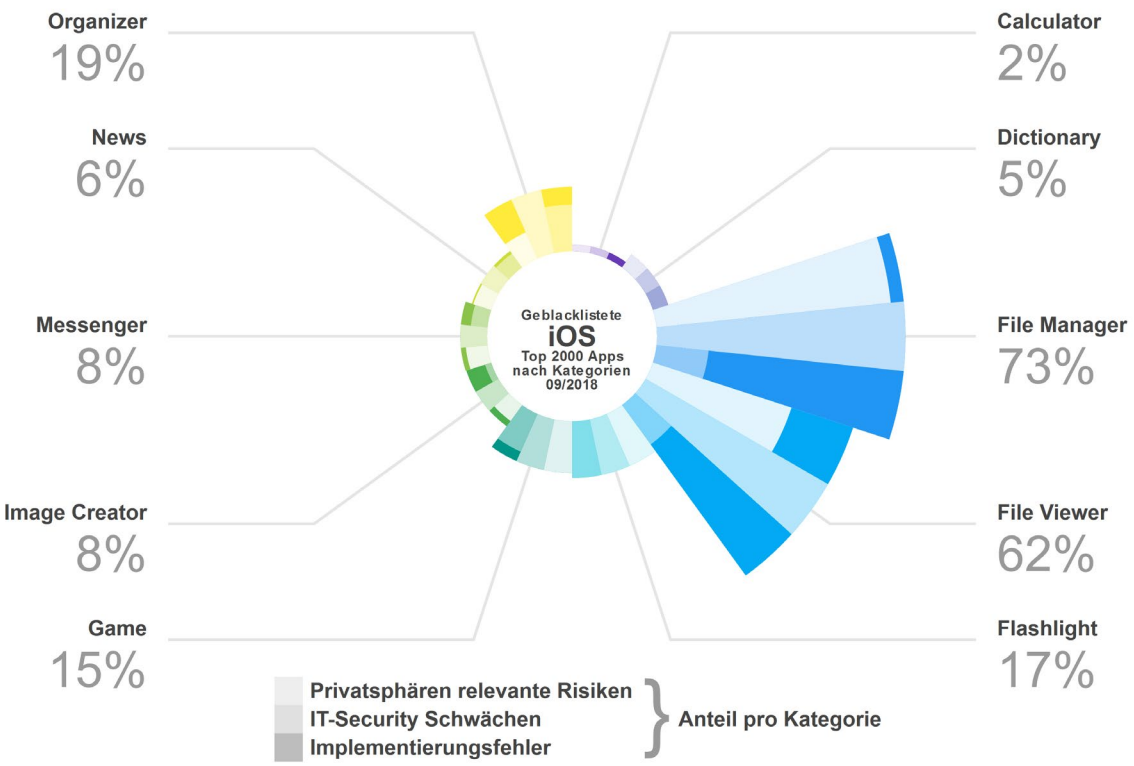


Abbildung 6: Anteil der geblacklisteten iOS-Apps je Funktionsklasse. Die Balken pro beispielhaft ausgewählter Funktionsklasse zeigen den jeweiligen Anteil der drei Risikoklassen. Bei den geblacklisteten 17% der Taschenlampen-Apps treffen jeweils alle Risken zu. (Appicaptor, September 2018)

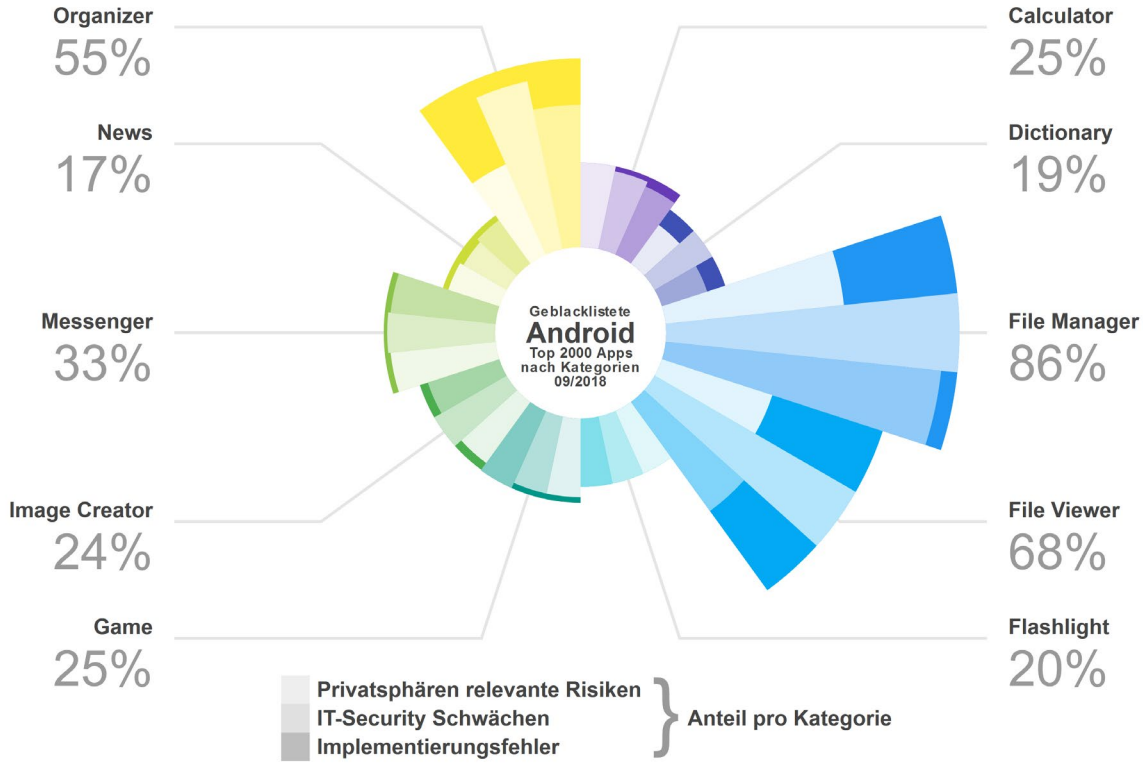


Abbildung 7: Anteil der geblacklisteten Android-Apps je Funktionsklasse. (Appicaptor, September 2018)

Unternehmensapps mit mehr Risiken

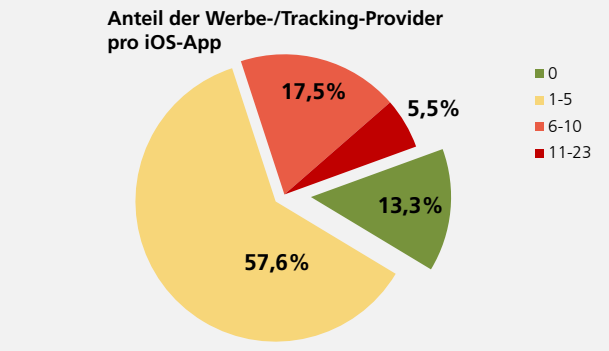
Für die Analyse der Apps durch Appicaptor werden die vollständigen App-Binärdaten automatisiert aus den App-Märkten heruntergeladen und in eine Zwischensprache zurückübersetzt. Für den vorliegenden Bericht stammen die Apps aus den deutschen App-Märkten. Nach dem Download erfolgt eine Vielzahl automatisierter Tests und Analysen, für die das Appicaptor-Framework Werkzeuge auf unterschiedlichen Ebenen zur Verfügung stellt, mit dem weiterhin die Einzelbefunde zu konsistenten Ergebnissen korreliert werden. Für den vorliegenden Bericht wurden statische Analyseverfahren verwendet. Damit wird sichergestellt, dass der gesamte App-Code untersucht wird. Es erfolgt damit eine gewisse Überapproximation, die alle Möglichkeiten der App bewertet, unabhängig davon, welche Bedingungen zum Auslösen einer Funktion notwendig sind.

Apps im Kategorievergleich

Bei der Bewertung der Tauglichkeit für den Unternehmenseinsatz zeigt sich, dass Apps für die Verarbeitung von Unternehmensdaten durchaus kritisch zu betrachten sind. Insbesondere die Funktionsklasse der File-Manager-Apps zeigt mit 73% als für Unternehmen ungeeignet eingestufte iOS-Apps ein deutliches Einsatzrisiko (siehe Abbildung 6). Dieses liegt bei Android mit 86% sogar noch etwas höher (siehe Abbildung 7). Die Gründe für die Blacklistung sind bei beiden Plattformen ein sehr hoher Anteil an IT-Security-Schwächen und privatsphärenrelevanter Risiken (siehe Kasten: Privatsphärenrisiken). Bei Android kommt dazu noch ein höherer Anteil an Implementierungsfehlern, durch die Angriffe erleichtert werden. Aufgrund der Zugriffsrechte auf Dateien und die häufig anzutreffenden Kommunikationsfunktionen sollten Unternehmen im Hinblick auf diese Analyseergebnisse bei File-Manager-Apps besonders auf die Einhaltung der Unternehmensrichtlinien achten.

Auch bei File-Viewer-Apps zeigt die Appicaptor-Analyse der Risiken Handlungsbedarf bei der geeigneten App-Auswahl. Diese Apps zeigen im Unternehmenskontext häufig vertrauliche Informationen in Form von Dokumenten an, die bei 62% der File-Viewer-Apps durch detektierte IT-Security-Schwächen von unautorisiertem Zugriff bedroht sind. Das Risiko bezieht sich dabei bspw. auf fehlenden Schutz in Lost-Device-Szenarien, Zugriff während der Kommunikation oder den Einsatz ungeeigneter oder zu schwacher Kryptografie.

Bei Organizer-Apps besteht insbesondere bei den kostenlosen Android-Apps aus dem Top-2.000-Katalog ebenfalls ein hohes Risiko für Unternehmen. Auch mit diesen Apps werden häufig Unternehmensdaten wie Kundenbeziehungen und -kontakte verarbeitet, die durch schlechte Sicherheitsqualität bei 55% der Android-Organizer-Apps leicht in die falschen Hände gelangen können. Bei iOS trifft dies auf 19% der analysierten Organizer-Apps zu.



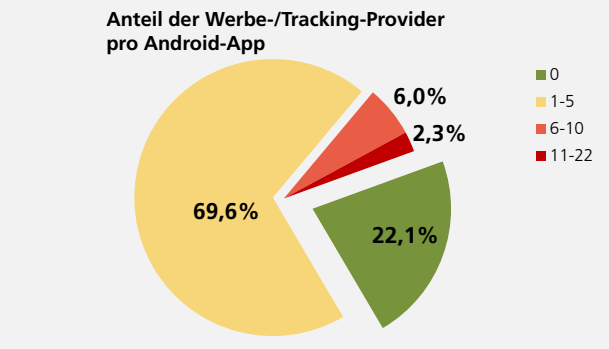


Abbildung 8: Anteil der Werbe- und Trackingnetzwerke pro iOS-/Android-App in den Top 2.000 der kostenlosen Apps der jeweiligen Plattform. In der Mehrzahl der Apps wurden bis zu 5 Werbe- und Trackingnetzwerke gefunden. Allerdings wurden auch Apps gefunden, die mit bis zu 23 solcher externen Dienstleister in Verbindung stehen. (Appicaptor, September 2018)

Da bei der Entscheidung für oder gegen die Unternehmenstauglichkeit einer App bei der Appicaptor-Analyse jeweils die Risiken von Verwundbarkeiten anhand der Zugriffsrechte, der verarbeiteten Daten und des App-Typs erfolgt, ist nachvollziehbar, dass in App-Typen wie News oder Taschenrechner durch die Standardregeln wesentlich weniger geblacklistet werden. Hier werden insbesondere Apps geblacklistet, die durch ihren hohen Anteil an Privatsphärerisiken auffallen oder aufgrund ihrer Implementierung Zugriff auf Unternehmensdaten gewähren können.

Der Vergleich zwischen den Plattformen zeigt, dass unter iOS Schwächen von Apps weniger häufig zur konkreten Risiken werden, da die iOS-Sandbox den Zugriff auf gemeinsame Ressourcen etwas besser schützt. So ist dort das Risiko geringer, dass eine verwundbare App auf alle gespeicherten Daten zugreifen könnte, wie es bei Android der lesende Zugriff auf die SD-Karte ermöglicht. Auch das automatisierte Auslesen und Versenden von SMS/E-Mails wird durch die ausschließliche Unterstützung über Dialoge, die durch den

Nutzer zu bestätigen sind, besser vor Missbrauch geschützt. Zudem wurden inzwischen auch Maßnahmen in Form von sogenannten Entitlements getroffen, die die unautorisierte Nutzung von nicht dokumentierten Schnittstellen mit direktem Zugriff verhindern.

Mitteilsame Apps

In diesem Jahr zeigt sich kaum ein Unterschied bei den Werbe- und Trackingnetzwerken im Vergleich zum Vorjahr. Der Anteil der Apps ohne detektierte Interaktion mit diesen Providern ist bei den iOS-Apps erneut geringfügig auf aktuell 13,3% gesunken (2017: 14,6%, 2016: 15,5%). Der Anteil von Apps mit mehr als 5 Werbe- und Trackingnetzwerken ist auf 23% gesunken (2017: 25,5%, 2016: 26,1%). Bei Android liegt der Anteil ohne detektierte Interaktion mit diesen Providern aktuell bei 22,1% (2017: 22,5%, 2016: 22%). Der Anteil der Apps mit mehr als 5 Providern ist bei Android von 10,3% wieder etwas auf 8,3% gesunken (2017: 10,3%, 2016: 4,6%).

Durch die gestiegene Nutzung der Werbe- und Trackingnetzwerke in Apps erhalten immer mehr Provider Einblick in das App-Nutzungsverhalten der Anwender, und pro Provider können mehr Details der Nutzer gesammelt werden. Bei einer geschäftlichen Nutzung können somit auch immer mehr Rückschlüsse über aktuelle Abläufe im Unternehmen gezogen werden. Dies resultiert daraus, dass ein Provider unter Einbeziehung von eindeutigen IDs, dem App-Kontext und der App-Aufrufhistorie konkrete Rückschlüsse auf den Ablauf im Unternehmen ziehen kann. Da ein Großteil der Kommunikation mit den Werbe- und Trackingnetzwerken immer noch unverschlüsselt stattfindet, stehen diese Informationen aber auch allen anderen zur Verfügung, die Zugriff auf das genutzte Netzwerk haben.

Fazit

Sowohl Apple als auch Google verbessern stetig die Sicherheit ihrer Smartphone-Plattformen und unterstützen Entwickler durch gute Standardvorgaben für die Entwicklung sicherer Apps. Insbesondere beim Thema Transportverschlüsselung zeigt diese Initiative aber noch immer nicht viel Wirkung, da viele Entwickler die sicheren Standardeinstellungen deaktivieren. Teilweise sicherlich aufgrund einer nicht beeinflussbaren fehlenden Unterstützung bei der Server-Gegenstelle, aber häufig auch aufgrund von Unwissenheit hinsichtlich sicherer Alternativen. Zudem nehmen die Risiken für Unternehmen durch die steigende Verwendung von unsicheren Hybrid-Apps weiter zu.

Unabhängig von der Plattform zeigen daher die Appicaptor-Ergebnisse: Die App-Auswahl stellt eine wichtige Entscheidung für die Unternehmenssicherheit dar. Gerade für dienstliche Abläufe sollten nur Apps zum Einsatz kommen, die den Unternehmensanforderungen entsprechen. Leider sind die Kriterien für Sicherheitsqualität aus den App-Märkten nicht ersichtlich und deren Prüfung nicht ausreichend.

Erst eine gezielte Schwachstellensuche hilft, die einzelnen App-Schwächen zu erkennen. Wie im vorliegenden Bericht verdeutlicht, ist die Bewertung der daraus resultierenden Risiken der entscheidende Faktor bei einer Entscheidung für oder gegen die App. Die Bewertung für ein ggf. notwendiges Untersagen der Nutzung im Unternehmen ist somit erst unter Einbeziehung von den verwendeten Daten, Zugriffsrechten, App-Funktion und -Typ möglich. Andernfalls sinkt die Nutzerakzeptanz für das Blacklisting, oder die Regeln müssen aufgrund dieser fehlenden Einbeziehung des App-Kontextes für alle Apps so gelockert werden, dass eigentlich kritische Apps dann auch nicht mehr geblacklistet werden.

Aufgrund der vielen App-Updates erfordert die Schwachstellensuche eine hohe Automatisierung, um die Analyseergebnisse für jeweils aktuelle Version zeitnah vorliegen zu haben. Mit den so gewonnen Informationen kann dann ein App-Freigabekonzept[11] umgesetzt werden, das die Freigabeanfragen und automatisiert erhobenen Analysedaten vollautomatisch durch das vorhandene Mobile Device Management System (MDM) oder Enterprise Mobility Management System (EMM) verarbeiten lässt. Alternativ unterstützen die Analyseergebnisse aber auch einen manuellen Freigabeprozess, der die analysierte Sicherheitsqualität der Apps für den Entscheidungsprozess nutzt.

[11] https://cordova.apache.org/docs/en/latest/guide/appdev/whitelist/index.html

Appicaptor: Analyse der Sicherheitsqualität

Appicaptor, eine Entwicklung des Fraunhofer SIT, scannt im Unternehmensauftrag automatisch große Mengen von beliebigen iOS- und Android-Apps, untersucht sie auf IT-Sicherheit und Einhaltung von Datenschutz-Vorgaben und bewertet, ob sie für den Business-Betrieb geeignet sind oder nicht. Dabei arbeitet Appicaptor wahlweise mit Standardregeln (Black- und Whitelisting) oder gibt Empfehlungen entsprechend den individuellen Sicherheitsvorgaben von Unternehmen.

Aufgrund der Automatisierung können die Tests wöchentlich wiederholt werden, sodass auch Änderungen bei sehr häufig aktualisierten Apps stets berücksichtigt werden.

Weitere Informationen zur Nutzung finden Sie unter: **https://www.appicaptor.de**

Analysierte App-Auswahl

Die Auswahl der analysierten Apps erfolgte anhand der Einstufung der Beliebtheit durch die App-Märkte. Es wurden pro Plattform jeweils die beliebtesten 200 Apps der folgenden 10 Kategorien aus den deutschen App-Märkten monatlich mit Appicaptor analysiert.

Apple App Store-Kategorien: Productivity, Utilities, Business, Socialnetworking, Finance, News, Lifestyle, Entertainment, Travel, Weather

Google Play Store-Kategorien: Productivity, Tools, Business, Social, Finance, News and Magazines, Lifestyle, Entertainment, Travel and Local, Communication

Privatsphärenrisiken

In dieser Analyse werden Befunde in Bezug auf eine unangemessene Preisgabe von Benutzerinformationen als Privatsphärenrisiken klassifiziert. Im Folgenden werden einige Beispiele erläutert:

Werbung/Tracking: Die App benutzt mehr als fünf Werbe- und Trackingnetzwerke und verbreitet dadurch persönliche Daten in Kombination mit dem Kontext der App-Nutzung.

Nicht plausibler Sensorzugriff: Die Nutzung von Smartphone-Sensoren (z.B. Mikrofone, GPS, Kamera, etc.) außerhalb des für den detektierten App-Typ üblichen Gebrauchs birgt ein Risiko bezüglich des Zugriffs auf sensible persönliche Daten.

Informationspreisgabe: Die Offenlegung von Standort-Daten oder Informationen über Web-Suchanfragen durch ungeschützte Kommunikation mit einem Dienstleister (z.B. Google), sodass neben dem Dienstleister auch Dritte die übermittelten Informationen einsehen können.

Benutzer-Identifikation: Die App versucht Zugang zu Informationen zu erlangen, anhand welcher ein Benutzer eindeutig identifiziert werden kann, wie beispielsweise Telefonnummer oder eindeutige Gerätenummern. Der Zugriff widerspricht dabei der App-Beschreibung (z.B. Taschenlampe).

Nutzung undokumentierter APIs (iOS): Die App beinhaltet Code, der Funktionen unveröffentlichter bzw. undokumentierter APIs aufruft. Diese Art der Programmierung beinhaltet das Risiko, dass diese APIs kritische Funktionen bereitstellen, die lediglich von Apps von Apple Inc. genutzt werden sollten. Apple erklärte, dass Apps, die solche APIs benutzen, vom Apple App Store ausgeschlossen werden. Dennoch findet Appicaptor regelmäßig Apps mit dieser Funktionalität im Apple App Store.

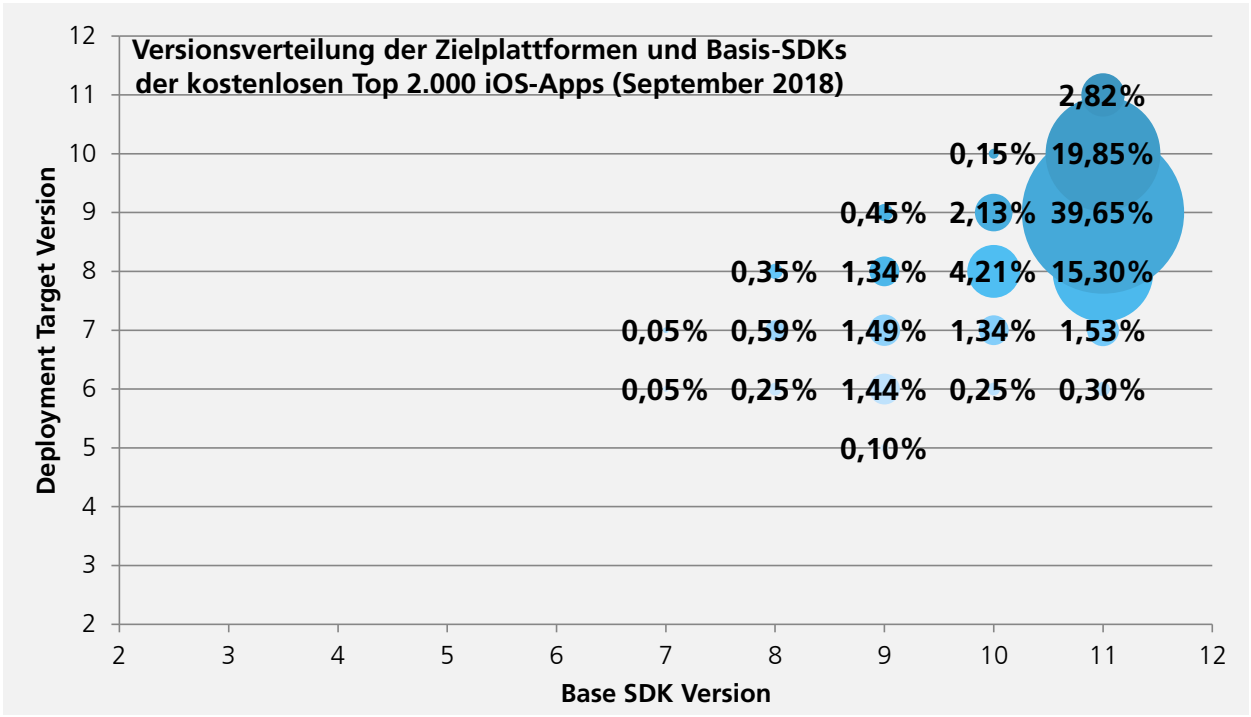


Abbildung 9: 2018 wurden deutlich weniger Apps mit älteren SDK-Versionen kleiner als SDK 9 erstellt. (Appicaptor, September 2018)

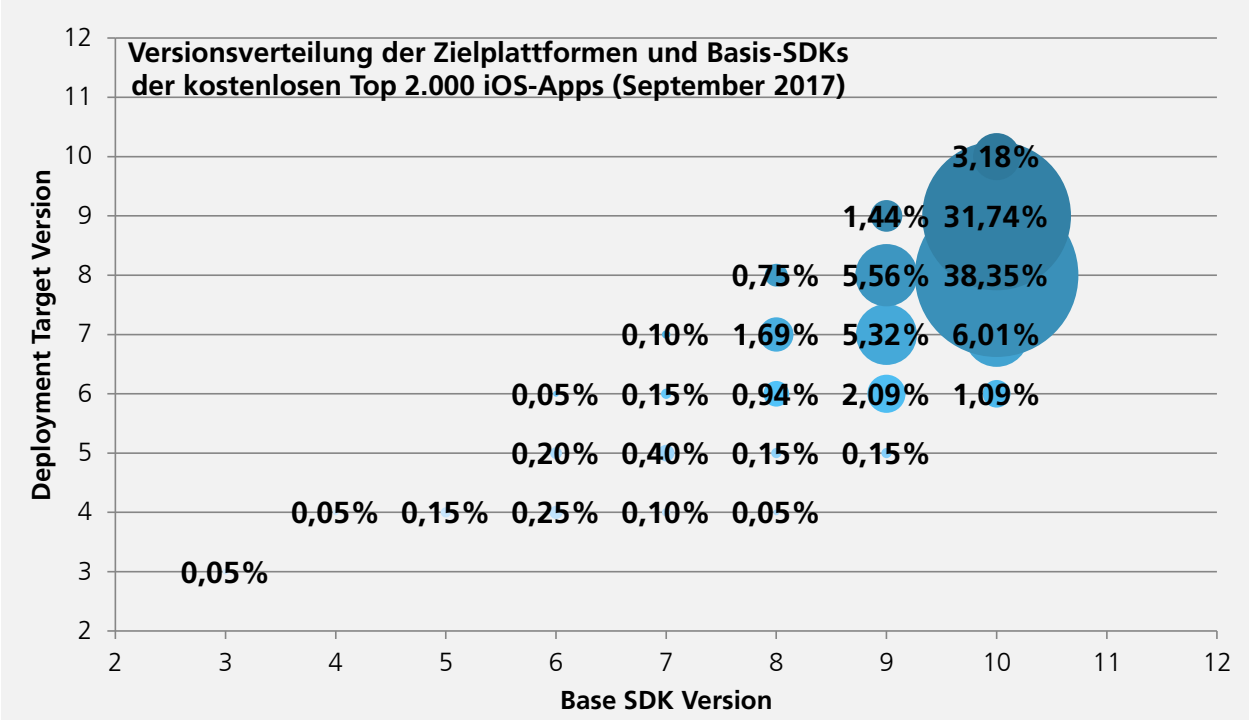


Abbildung 10: 4,3% der iOS-Apps wurden mit SDK-Versionen kleiner 9 erstellt, 2018 waren es nur noch 1,3%. (Appicaptor, September 2017)

Kontakt
Dr. Jens Heider

Fraunhofer SIT
Rheinstrasse 75
64295 Darmstadt, Germany

Telefon 06151 869-233
appicaptor@sit.fraunhofer.de
https://www.appicaptor.de