بسمه‌تعالی

آیا ابزارهای رایگان تحلیل امنیت اندروید در کشف آسیب‌پذیری‌های شناخته شده موثر عمل می‌کنند؟

گزارش فنی

شناسه سند ..... TR_AndroidFreeAnalysis_13980427
نوع سند ..... گزارش فنی
شماره نگارش ۱
تاریخ نگارش ۱۳۹۸/۰۴/۲۷
طبقه‌بندی سند ..... **عادی**

شاهرود، میدان هفت تیر، بلوار دانشگاه، دانشگاه صنعتی شاهرود، مرکز تخصصی آپا، کد پستی: ۳۶۱۹۹۹۵۱۶۱

(۰۲۳) ۲۲۰۴-۳۲۳۹ (۰۲۳) ۰۲۵۱-۳۲۳۰ cert.shahroodut.ac.ir

فهرست مطالب

فهرست جداول

چکیده

افزایش توجه و تمایل به مبحث امنیت اندروید، موجب شد تا تلاش‌های گسترده‌ای در راستای کمک به توسعه‌دهندگان برای ساخت برنامه‌های امن‌تر صورت گیرد. حاصل این تلاش‌ها، ابزارها و تکنیک‌هایی بود که قابلیت کشف آسیب‌پذیری (و رفتارهای مخرب) در برنامه‌های اندرویدی را دارا بودند. با این حال تاکنون هیچ ارزیابی از میزان موثر بودن این ابزارها و تکنیک‌ها در کشف آسیب‌پذیری‌های شناخته شده انجام نشده است. به همین منظور، مرجع [1] میزان موثر بودن ابزارهای کشف آسیب‌پذیری را مورد ارزیابی قرار داده است. این ارزیابی به توسعه‌دهندگان برنامه‌های اندرویدی در انتخاب ابزار تحلیل امنیت مناسب کمک می‌کند. در این گزارش خلاصه‌ای از مقاله [1] ارائه می‌کنیم.

کلمات کلیدی: آسیب‌پذیری، امنیت اندروید، مخزن جیرا، ابزارهای تحلیل امنیت

۱ مقدمه

با اینکه فروشگاه‌های برنامه‌های اندرویدی[۱] تلاش می‌کنند تا برنامه‌های مخرب را از اکوسیستم دور نگه‌دارند، برنامه‌های مخرب می‌توانند از طرق مختلف (نصب برنامه از منابع نامطمئن، ناتوانی در کشف رفتارهای مخرب در برنامه‌ها، دسترسی به وب‌سایت‌های مخرب) وارد اکوسیستم شوند؛ بنابراین نیاز است توسعه دهندگان برنامه‌های خود را امن کنند.

ارزیابی‌هایی که تاکنون صورت گرفته توسط عوامل مختلفی محدود شده‌اند؛ مانند:

- تنها ابزارهای آکادمیک بررسی شده‌اند.
- تعداد کمی از ابزارهای امنیتی بصورت عملی بررسی شده‌اند.
- بر اساس معیار[۲]ی بررسی شده‌اند که نمود واقعی رخداد آسیب‌پذیری در دنیای واقعی نیستند.

---

[۱] App store

[۲] benchmark

۲ آیا معیارها، آسیب‌پذیری‌ها را دقیقا همانطوری که در دنیای واقعی رخ می‌دهد، نشان می‌دهند؟

برای ارزیابی ابزارهای امنیتی، کاتالوگ آسیب‌پذیری‌های برنامه‌های اندرویدی به نام مخزن جیرا[۳] - یک مخزن رو به رشد از آسیب‌پذیری‌های برنامه‌های اندرویدی - را بررسی کردیم. هر معیار در این مخزن، یک آسیب‌پذیری مشخص X را ضبط می‌کند که شامل سه برنامه اندرویدی به شرح زیر است:

۱. برنامه آسیب‌پذیر[۴] که آسیب‌پذیری X را دارد.

۲. یک برنامه مخرب[۵] که می‌تواند از آسیب‌پذیری X در برنامه آسیب‌پذیر استفاده کرده و به برنامه صدمه بزند.

۳. یک برنامه امن که آسیب‌پذیری X را ندارد.

این مخزن شامل ۴۲ معیار است که در ۷ گروه دسته‌بندی می‌شوند:

۱. دسته رمز: آسیب‌پذیری‌هایی که به رمزنگاری مربوط می‌شوند؛ مثال: حاصل رمزنگاری مجدد یک پیام رمزنگاری شده در مود EBC، خود پیام است.

۲. دسته ICC[۶]: آسیب‌پذیری‌هایی که درنتیجه تعاملات قسمت‌های داخلی مختلف یک برنامه‌ی اندرویدی (فعالیت‌ها، سرویس‌ها، دریافت‌کننده پیام‌های عمومی[۷]، ارائه‌دهندگان محتوا[۸]) ایجاد می‌شود.

۳. شبکه‌سازی: برنامه‌های اندرویدی می‌توانند از طریق کتابخانه‌های شبکه‌سازی یا به کارگیری پروتکل‌های مختلف (غیر از وب) با هم ارتباط برقرار کنند که خود می‌تواند آسیب‌پذیری ایجاد کند؛ بطور مثال می‌توانند برای شنیدن اتصالات کاربر، سوکت‌های سرور را باز کنند که ممکن است موجب درز اطلاعات شود.

۴. مجوزها[۹]: علاوه بر مجوزهای تعریف شده توسط خود سیستم، برنامه‌های اندرویدی هم می‌توانند مجوزهایی تعریف و استفاده کنند. این مجوزها می‌توانند با چهار سطح حفاظت موجود (عادی، خطرناک، امضاء، امضا یا سیستم) ترکیب شوند تا دسترسی به

---

[۳] Ghera repository : https://bitbucket.org/secure-it-i/android-app-vulnerability-benchmarks/src/master/

[۴] Benign app

[۵] (exploit) malicious app

[۶] Inter Component Communication

[۷] broadcast receives

[۸] Content provider

سرویس‌ها و ویژگی‌های مختلف را کنترل کنند. مثال: مجوزهایی با سطح محافظت عادی، بطور خودکار، امکان درخواست کردن برای دریافت مجوز، در حین نصب را دارند درنتیجه هر مولفه یا واسطی که توسط مجوزهای عادی محافظت شود قابل دسترسی برای هر برنامه‌ی نصب شده‌ای خواهد بود.

۵. حافظه: شامل آسیب‌پذیری‌های مربوط به پایگاه داده وحافظه است.

۶. سیستم: شامل آسیب‌پذیری‌های مربوط به کتابخانه‌های سیسمتی که با فرایندها سروکار دارند، است.

۷. وب: برنامه‌های اندرویدی می‌توانند از طریق کتابخانه‌های وب با وب‌سرورها ارتباط SSL/TLS و یا بدون آن برقرار کنند، که خود می‌تواند آسیب‌پذیری ایجاد کند؛ بطور مثال برنامه‌هایی که به سرورهای ریموت از طریق پروتکل http متصل می‌شوند خطر حمله مرد میانی[۱۰] را به همراه دارد.

آسیب‌پذیری‌هایی که مبنای ارزیابی ابزارها هستند باید معتبر (به واقع یک ضعف مشخص در برنامه ایجاد کند)، کلی (قابل استفاده در حالت کلی بدون احتیاج به دسترسی آدمین و یا روت بودن دستگاه)، قابل اعمال (بتوان آسیب‌پذیری را به نحوی اعمال کرد که به برنامه صدمه وارد کند) و جاری (وجود آسیب‌پذیری در برنامه‌های جاری و آینده) باشند.

از آنجایی که آسیب‌پذیری‌های موجود در جیرا در مقالات اخیر و مستندات اندروید وجود دارد پس معتبرند، برنامه آسیب‌پذیر و برنامه مخرب موجود در هر معیار می‌توانند بر روی شبیه‌سازها و یا دستگاه‌های اندرویدی اجرا شوند پس قابل اعمال و کلی نیز هستند همچنین این آسیب‌پذیری‌ها بر اساس کتابخانه‌های اندرویدی سطح ۱۹ تا ۲۵ (بجز ۲۰) - که شامل ۹۰درصد از دستگاه‌های اندرویدی جهان هستند - می‌باشند، پس جاری نیز هستند.

در ادامه برای ارزیابی عملکرد ابزارها در کشف آسیب‌پذیری برنامه‌ها، به نمونه‌برداری تعدادی از برنامه‌های اندرویدی از Androzoo[۱۱] - بعنوان منبعی از برنامه‌های اندرویدی در دنیای واقعی - پرداختیم. Androzoo لیستی از همه‌ی apkهای موجود را به همراه هش SHA256 آنها، حجم و تاریخ هر کدام، نگهداری می‌کند.

تصمیم بر آن شد که تنها apkهایی که سطح کتابخانه‌ی هدف آنها ۱۹ یا بالاتر و حداقل سطح کتابخانه‌شان ۱۴ یا بالاتر بود انتخاب شوند. برای هر apk براساس سطح کتابخانه‌ای که استفاده کرده بود یک پروفایل کتابخانه ایجاد کردیم که شامل صفات فایل manifest، متدها و فیلدهای استفاده شده در برنامه و کتابخانه‌های مرتبط است.

---

[۹] permissions

[۱۰] Man-in-the-Middle

[۱۱] https://androzoo.uni.lu/

نتیجه بررسی‌های انجام شده نشان می‌دهد که تعداد زیادی از برنامه‌ها در دنیای واقعی از تعداد زیادی از کتابخانه‌های موجود در معیار جیرا استفاده می‌کنند. بنابراین می‌توان نتیجه گرفت جیرا نمودی از برنامه‌های اندرویدی دنیای واقعی است.

۳ راهکارهای تحلیل امنیت برنامه‌های اندرویدی

در حین ایجاد مخزن جیرا، از راهکارهای مختلفی برای تامین امنیت برنامه‌های اندرویدی مطلع شدیم و شروع به جمع‌آوری اطلاعات درباره‌ی آنها کردیم.۶۴ راهکار امنیتی مربوط به اندروید را بررسی کردیم که در ابعاد زیر دسته‌بندی شدند:

۱. ابزارها در مقابل چارچوب‌ها[۱۲]: ابزارها سریع و در دسترس‌ترند و یک مجموعه ثابتی از آسیب‌پذیری‌ها را پوشش می‌دهند درحالیکه در چارچوب‌ها می‌توان ابزارهایی برای کشف مسائل مشخص امنیتی تولید کرد و مجموعه‌ی نسبتا بیشتری از آسیب‌پذیری‌ها را پوشش می‌دهد.

۲. رایگان در مقابل تجاری

۳. ماندگار درمقابل غیرماندگار: راهکارهای غیرماندگار نسخه‌های کنونی اندروید را پشتیبانی نمی‌کنند.

۴. کشف آسیب‌پذیری درمقابل کشف رفتار مخرب: عموماً توسعه‌دهندگان از اولی و فروشگاه‌های برنامه‌های اندرویدی و کاربران از دومی استفاده می‌کنند.

۵. تحلیل ایستا درمقابل تحلیل پویا: راهکارهای امنیتی مذکور می‌توانند مبتنی بر هرکدام از این دو تحلیل باشند. تحلیل ایستا به تجزیه و تحلیل در محیط اجرا، معنای برنامه[۱۳]، تعاملات با دیگر برنامه‌ها و کاربر می‌پردازد. در مقابل، تحلیل پویا به جستجوی مشکلات امنیتی در زمان اجرا[۱۴] می‌پردازد.

۶. محلی درمقابل کنترل از راه دور: راهکارهای محلی، روی سیستم نصب می‌شوند و بصورت محلی توسط توسعه‌دهندگان اجرا می‌شوند. راهکارهای کنترل از راه دور از طریق سرویس‌های وب در دسترس هستند و توسط توسعه‌دهندگان همان راهکار، نگهداری می‌شود؛ به‌این‌صورت که توسعه‌دهندگان فایل apk خود را به ابزار می‌دهند و یک گزارش از تحلیل دریافت می‌کنند؛ در حین استفاده از این نوع ابزارها توسعه‌دهندگان هیچ اطلاعی از اینکه چه اتفاقی برای فایل apk می‌افتد، ندارند.

---

[۱۲] Frameworks

[۱۳] Program semantics

[۱۴] Run time

۴ کدام ابزارها را بررسی کنیم؟

از بین ۶۴ راهکار و ابزار بررسی شده، ۵ راهکار به دلیل نداشتن مستندات، نبود دستورالعمل، انگلیسی نبودن مستندات و یا نقص در آن‌ها، ۶ ابزار به دلیل بررسی کردن سیاست‌های امنیتی بجای آسیب‌پذیری، AppRay به دلیل تجاری بودن، AndroWarn و ScanDroid به دلیل به روز نبودن (آخرین آپدیت در ۲۰۱۳)، TaintDroid را به دلیل اینکه فقط از سطح کتابخانه‌ی ۱۸ و پایین‌تر پشتیبانی می‌کرد، شش ابزار به دلیل بررسی رفتارهای مخرب بجای آسیب‌پذیری، دو ابزار به دلیل سازگار نبودن با مخزن جیرا و چهار ابزار ریموت حذف شدند و درنهایت ۱۴ ابزار برای ارزیابی باقی ماندند. اسامی و مشخصات ۱۴ ابزار در جدول ۱ آمده است.

جدول ۱. معرفی ۱۴ ابزاری که مورد ارزیابی قرار گرفته اند

نام ابزار	نسخه/شناسه commit	به روز رسانی [انتشار]	S/D	L/R	A/N	H/E	زمان صرف شده برای راه‌اندازی(ثانیه)
Amandroid	3.1.2	2017[2014]	S	L	A	E	3600
AndroBugs	$7fd3a2c$	2015[2015]	S	L	N	H	600
AppCritique	?	? [?]	?	R	N	?	
COVERT	2.3	2015[2015]	S	L	A	E	2700
DevKnox	2.4.0	2017[2016]	S	L	N	H	600
DIALDroid	25daa37	2018[2016]	S	L	A	E	3600
FixDroid	1.2.1	2017[2017]	S	L	A	H	600
FlowDroid	2.5.1	2018[2013]	S	L	A	E	9000
HornDroid	aa92e46	2018[2017]	S	L	A	E	600
JAADAS	0.1	2017[2017]	S	L	N	H/E	900
MalloDroid	78f4e52	2013[2012]	S	L	A	H	600
Marvin-SA	6498add	2016[2016]	S	L	N	H	600
MobSF	B0efdc5	2018[2015]	SD	L	N	H	1200
QARK	1dd2fea	2017[2015]	S	L	N	H	600

برای هر ابزار مواردی مانند، مبتنی بر تحلیل ایستا (S) یا پویا (D) یا هر دو (SD) بودن، از راه دور اجرا شدن (R) یا محلی بودن(L)، آکادمیک بودن (A) یا آکادمیک نبودن (N)، استفاده از تحلیل عمیق (E) یا تحلیل سطحی[۱۵] (H) و زمان صرف شده برای راه‌اندازی آن ابزار در سیستم لینوکس (Set Up Time) گزارش کردیم.

تحلیل سطحی به جستجوی کد اسمل‌ها[۱۶] (معمولا خطا نیستند، از لحاظ فنی غلط نیستند و عملکرد برنامه را مختل نمی‌کنند ولی نشان‌دهنده‌ی ضعف در طراحی هستند که موجب افزایش ریسک خطاها و شکست‌ها در آینده می‌شود) و الگوهاست اما تحلیل عمیق به

[۱۵] Shallow analysis

بررسی جریان‌های داده می‌پردازد. همچنین اینکه هر ابزار قابلیت کشف کدام دسته از آسیب‌پذیری‌ها (در بخش ۲. مطرح شد) را دارد، نیز در جدول ۲ آمده است.

جدول ۲. قابلیت کشف هر دسته از آسیب پذیری‌ها در هر یک از ابزارها

	دسته بندی آسیب‌پذیری‌ها						
ابزارها	رمز	ICC	شبکه	مجوز	حافظه	سیستم	وب
Amandroid	✓	✓	✓		✓		✓
AndroBugs	✓	✓	✓	✓	✓	✓	✓
AppCritique	✓	✓	✓	✓	✓	✓	✓
COVERT	✓	✓	✓	✓	✓	✓	✓
DevKnox	✓	✓	✓	✓	✓	✓	✓
DIALDroid	✓	✓	✓	✓	✓	✓	✓
FixDroid	✓	✓	✓	✓	✓	✓	✓
FlowDroid		✓	✓		✓		
HornDroid	✓	✓	✓		✓		✓
JAADAS	✓	✓	✓	✓	✓	✓	✓
MalloDroid							✓
Marvin-SA	✓	✓	✓	✓	✓	✓	✓
MobSF	✓	✓	✓	✓	✓	✓	✓
QARK	✓	✓	✓	✓	✓	✓	✓

[۱۶] Code smells

از آنجایی که ابزار MalloDroid بر روی کشف آسیب‌پذیری‌های مربوط با SSL/TLS تمرکز دارد تنها در مبحث SSL/TLS مربوط به آسیب‌پذیری‌های دسته وب، مورد ارزیابی قرار گرفت.

۴-۱ معرفی مختصر ابزارهای انتخابی

- چارچوب Amandroid: شامل هفت زیرپکیج برای کشف آسیب‌پذیری‌ها است؛ $Amandroid_1$ (مربوط به درز اطلاعات)، $Amandroid_2$ (تزریق intent)، $Amandroid_3$ (درز دستور[۱۷])، $Amandroid_4$ (ردیابی پسورد)، $Amandroid_5$ (ردیابی AOuth[۱۸])، $Amandroid_6$ (سوء استفاده از SSL)، $Amandroid_7$ (سوء استفاده از رمز) که هر کدام می‌تواند بعنوان یک ابزار مستقل تحلیل بکار رود.
- چارچوبAndroBugs[۱۹]: یک سیستم تحلیل امنیت اندروید است و با اینکه رابط کاربری خیلی خوبی ندارد، اما ادعا می‌کند در کمتر از دو دقیقه هر اسکن را با دقت و به درستی انجام می‌دهد (قابل استفاده در سیستم عامل‌های ویندوز و لینوکس).
- COVERT[۲۰]: ابزاری برای تحلیل ترکیبی آسیب‌پذیری‌های inter-application[۲۱] است. این ابزار بصورت خودکار آسیب‌پذیری‌هایی که در اثر تعامل برنامه‌های تشکیل‌دهنده‌ی یک سیستم، ایجاد می‌شود را شناسایی می‌کند. در نتیجه این ابزار مشخص می‌کند آیا نصب که یک گروه مشخص از برنامه‌ها (با مجوزها و تعاملاتشان) کنار یکدیگر امن است یا خیر. این ابزار فایل apk برنامه را بعنوان ورودی دریافت می‌کند.
- DevKnox[۲۲]: یک پلاگین در اندروید استودیو است که به برنامه‌نویسان کمک می‌کند تا درحین برنامه‌نویسی، مشکلات امنیتی برنامه خود را کشف و رفع کنند. نسخه‌ی رایگان Devknox Lite شامل ۳۰ تست امنیتی و نسخه‌ی تجاری آن شامل ۶۰ تست امنیتی و تحلیل ایستای برنامه است[۴].
- DIALDroid[۲۳]: ابزاری با مقیاس‌پذیری بالا برای شناسایی تصادم‌های بین برنامه‌ای (درون برنامه‌ای) و افزایش حق دسترسی[۲۴] در بین برنامه‌های اندرویدی است.

---

[۱۷] Command

[۱۸] a delegated authorization framework for REST/APIs. It enables apps to obtain limited access (scopes) to a user's data without giving away a user's password

[۱۹] https://github.com/AndroBugs/AndroBugs_Framework

[۲۰] http://seal.ics.uci.edu/projects/covert/

[۲۱] برون برنامه‌ای

[۲۲] https://devknox.io/

- FixDroid[۲۵]: یکی از پلاگین‌های اندورید استودیو است. از ویژگی‌های این پلاگین دادن هشدارهای امنیتی و توضیحات درباره‌ی آنها است.
- FlowDroid: ابزاری است که به تحلیل taint جریان داده در برنامه‌های اندرویدی و جاوا می‌پردازد. نخستین تحلیل کامل فیلد و context که در آن چرخه حیات برنامه و ویجت‌های رابط کاربری[۲۶] در نظر گرفته شده است[۲].

لازم به ذکر است که تحلیل taint، نوعی بخصوص از تحلیل جریان داده است که در آن، یک شی به اصطلاح " آلوده" از منبع (به همراه داده‌های آلوده مرتبط با آن) ردگیری می‌شود. این عملکرد موجب می‌شود که بتوانیم تاثیر شی آلوده را در حین اجرای برنامه ببینیم. این تحلیل برای یافتن منشا درز اطلاعات و آسیب‌پذیری‌های برنامه استفاده می‌شود. بطور دقیق‌تر شامل چک کردن متغیرهایی است که می‌تواند توسط کاربر تغییر داده شوند. همه‌ی داده‌های ورودی توسط کاربر، اگر به درستی بررسی نشوند می‌تواند خطرناک باشد.

- HornDroid: یک رویکرد تازه نسبت به تحلیل ایستا با استفاده از عبارات هورن[۲۷] ارائه می‌دهد[۳].
- JAADAS[۲۸]: یک چارچوب ارزیابی نقص پیشرفته برای برنامه‌های کاربردی اندروید است که به زبان جاوا و اسکالا نوشته شده است؛ از ویژگی‌های این چارچوب می‌توان به تحلیل سوءاستفاده از کتابخانه، دریافت چند فایل dex.[۲۹] و تحلیل همه‌ی آنها با یکدیگر، تحلیل DoS[۳۰] داخلی (از کار افتادن intentها) و تحلیل جریان داخلی روال‌ها اشاره کرد. این چارچوب تحلیل را در دو حالت تحلیل سریع (فقط تحلیل درون-روالی $JAADAS_H$) و تحلیل کامل (تحلیل درون و برون روالی $JAADAS_E$) انجام می‌دهد.
- MalloDroid[۳۱]: ابزاری است که اعتبار گواهی‌های SSL در برنامه‌های اندرویدی را بررسی می‌کند و در واقع یک ابزار کوچک ساخته شده در androguard (چارچوب مهندسی معکوس برای برنامه‌های اندرویدی) می‌باشد؛ بنابراین ابزار androguard نیاز به استفاده از MalloDroid دارد.

---

[۲۲] https://github.com/dialdroid-android/DIALDroid

[۲۳] Privilege escalation

[۲۴] https://plugins.jetbrains.com/plugin/9497-fixdroid

[۲۵] UI widgets

[۲۷] در منطق ریاضی عبارت هورن یک فرمول منطقی است که حداکثر یک لیترال (منطق ریاضی) مثبت داشته باشد.

[۲۸] https://github.com/flankerhqd/JAADAS

[۲۹] Dalvik Executable

[۳۰] denial-of-service

[۳۱] https://github.com/sfahl/mallodroid

- MobSF[۳۲]: چارچوب خودکاری که قادر به انجام تحلیل پویا، ایستا و رفتارهای مخرب است و می‌تواند برای تحلیل سریع امنیت برنامه‌های اندرویدی استفاده شود و با بکارگیری CapFuzz (کتابخانه‌ی وب مختص امنیت)، قابلیت فازینگ[۳۳] کتابخانه‌های وب را دارا است.
- Marvin-SA[۳۴]: یک اسکنر آسیب‌پذیری در برنامه‌های اندرویدی است که از تحلیل سطحی برای آنالیز استفاده می‌کند همچنین از ابزار androguard و چارچوب SAAF[۳۵] کمک می‌گیرد.
- QARK[۳۶]: می‌تواند کد اصلی و apk یک برنامه را تحلیل کند. این ابزار بایت کد dex. یک برنامه را گرفته و به کد اصلی تبدیل می کند. از آنجایی که ساختار کدی که مهندسی معکوس شده ممکن است با ساختار کد اصلی متفاوت باشد، عدم اطلاع از تاثیر این تفاوت در تشخیص آسیب‌پذیری توسط این ابزار باعث شد تا ارزیابی هم، با ورودی apk ($QARK_A$) و هم با ورودی کد اصلی ($QARK_S$) انجام شود. در این ابزار نیاز نیست که دستگاه مورد آزمایش روت باشد.

۵ نتیجه ارزیابی

نتایج حاصل از ارزیابی ابزارهای کشف آسیب‌پذیری در جدول ۳ آمده است.

[۳۲] https://github.com/MobSF/Mobile-Security-Framework-MobSF

[۳۳] Fuzzing، يك تكنيك تست نرم افزار است كه براي پيدا كردن خطاهاي كد و حفره‌هاي امنيتي در نرم افزار مورد استفاده قرار مي‌گيرد و شامل فراهم كردن داده‌هاى ناصحيح، غيرمنتظره، اعداد تصادفى به‌عنوان ورودى به يک نرم‌افزار است.

[۳۴] Marvin Static Analyzer (https://github.com/programa-stic/Marvin-static-Analyzer)

[۳۵] https://github.com/SAAF-Developers/saaf/

[۳۶] https://github.com/linkedin/qark

جدول ۳. نتایج حاصل از ارزیابی ابزارهای کشف آسیب پذیری

ابزار	رمز (٤)	ICC (١٦)	شبکه (٢)	مجوز (١)	حافظه (٦)	سیستم (٤)	وب (٩)	تعداد معیارهای قابل اعمال	تعداد معیارهای شناخته شده	برنامه آسیب‌پذیر TP	برنامه آسیب‌پذیر FN	برنامه امن TN	دیگر
$Amandroid_1$		۳/۹/۰/۷	۰/۱/۰/۱		۰/۱/٤/۰ {۱}		۰/۳/۰/٦	۱٥	۱۳	۱	۱٤	۱٤	۳
$Amandroid_2$		X			۰/۲/۰/۱ {۳}		۰/۰/۰/٥ {٤}	۲٤	۲۳	۰	۳	۳	۰
$Amandroid_3$		۲/۸/۰/۸	۰/۱/۰/۱				۰/۳/۰/٦	۱٤	۱۲	۰	۱٤	۱٤	۲
$Amandroid_4$		۲/۳/۰/۱۳						۳	۳	۰	۳	۳	۲
$Amandroid_6$							۰/۳/۰/٦	۳	۳	۰	۳	۳	۰
$Amandroid_7$	۰/۲/۲/۰							٤	٤	۲	۲	٤	۰
AndroBugs*	N	۳/۱٤/۲/۰	N	۰/۰/۱/۰	N	۰/۰/٤/۰	۱/٥/٤/۰	٤۲	۳۹	۱۱	۳۱	٤۲	٤
AppCritique*	۰/۲/۲/۰	N	N	N	۰/۳/۳/۰	N	۰/۷/۲/۰	٤۲	?	۷	۳٥	٤۲	۰
COVERT	N	N	۰/۱/۰/۱	N	N	N	۰/۱/۰/۸	۳۳	۳۰	۰	۳۳	۳۳	۰
DevKnox*	۰/۳/۱/۰	N	N	N	N	D	N	٤۲	٤۰	٥	۳۷	۳۸	۰
DIALDroid	N	N	۰/۱/۰/۱	N	N	N	۰/۱/۰/۸	۳۳	۳۳	۰	۳۳	۳۳	۰
FixDroid	۲/۲/۱/۰	۰/۰/۱/۱۳	N	N	۰/٤/۱/۰	۰/۰/٤/۰	۰/۷/۲/۰	۲٥	۲٥	۹	۱٦	-	۲
FlowDroid		N	N		N			۲٤	۲٤	۰	۲٤	۲٤	۰
HornDroid	N	۷/۱٥/۱/۰	N		۱/٦/۰/۰		۰/۷/۲/۰	۳۷	۳۷	۱	۳٦	۳۷	۹
$JAADAS*_H$	N	۰/۱٤/۲/۰	N	N	N	N	۱/٥/٤/۰	٤۲	٤۰	٦	۳٦	٤۲	۱
$JAADAS*_E$	N	۰/۱٤/۲/۰	N	N	N	N	۱/٥/٤/۰	٤۲	٤۰	٦	۳۸	٤۲	۱
Marvin-SA	۰/۳/۱/۰	۳/۱۱/٥/۰	N	۰/۰/۱/۰	۲/٦/۱/۰	۰/۰/٤/۰	۰/٥/٤/۰	٤۲	۳۹	۱٥	۲۷	٤۲	٥
MalloDroid			X				۰/۱/۰/٥	٤	٤	۰	۱	۱	۰

							[۳]						
MobSF*	۰/۳/۱/۰	۰/۱۱/۵/۰	N	۱/۰/۱/۰	۰/۵/۱/۰	۰/۰/٤/۰	۰/٦/۳/۰	٤۲	٤۲	۱۵	۲۷	٤۲	۱
QARK*$_A$	N	۰/۱۳/۳/۰	N	۰/۰/۱/۰	N	۰/۰/٤/۰	۰/۷/۲/۰	٤۲	٤۰	۱۰	۳۲	٤۲	۰
QARK*$_S$	N	۰/۱۳/۳/۰	N	۰/۰/۱/۰	N	۰/۰/٤/۰	۰/۷/۲/۰	٤۲	٤۰	۱۰	۳۲	٤۲	۰
تعداد کشف نشده‌ها	۱	۵	۲	۰	۲	۰	۲						

در جدول ۳، تعداد آسیب‌پذیری‌هایی (معیارهای) که در این ارزیابی، برای ابزار، قابل اعمال بود و تعداد آسیب‌پذیری‌هایی که در زمان ساخت آن ابزار یا آخرین به روزرسانی آن، شناخته شده بود به ترتیب در ستون‌های "تعداد معیارهای قابل اعمال" و "تعداد معیارهای شناخته شده" آمده است.

- اعداد داخل پرانتز نشان‌دهنده‌ی معیارهای موجود در آن دسته هستند بطور مثال ۳ معیار در زمینه رمز وجود دارد.
- خانه‌های خالی گویای این هستند که ابزار بر هیچ‌کدام از معیارهای(آسیب‌پذیری) آن دسته قابل اجرا نبوده است. N گویای این است که ابزار هم برنامه‌ی آسیب‌پذیر و هم برنامه‌ی امن را در همه‌ی معیارهای آن دسته غیرآسیب‌پذیر تشخیص داده است. X گویای ناتوانی ابزار در پردازش همه‌ی معیارهای آن دسته است و D گویای این است که ابزار هم برنامه آسیب‌پذیر و هم برنامه امن را در همه‌ی معیارهای آن دسته آسیب‌پذیر تشخیص داده است.
- در (H/I/J/K)، H نشان‌دهنده‌ی تعداد معیارهایی است که برای ابزار غیرقابل‌اجرا بوده است، I تعداد معیارهایی که در آن، ابزار برنامه آسیب‌پذیر را آسیب‌پذیر و برنامه امن را غیرآسیب‌پذیر تشخیص داده است، J تعداد معیارهایی است که در آن هر دو برنامه‌ی امن و آسیب‌پذیر را غیرآسیب‌پذیر تشخیص داده است و در K تعداد معیارهایی است که در آن در هر دو برنامه امن و آسیب‌پذیر هیچ آسیب‌پذیری را کشف نکرده است.
- تعداد معیارهایی که ابزار نتوانسته پردازش کند در [ ] و تعداد معیارهایی که ابزار در آنها هر دو برنامه‌ی امن و آسیب‌پذیر را آسیب‌پذیر تشخیص داده در { } آمده است.
- برنامه آسیب‌پذیر (true positive) TP: تعداد برنامه‌های آسیب‌پذیری که ابزار به درستی آسیب‌پذیر تشخیص داده است.
- برنامه آسیب‌پذیر (false negative) FN: تعداد برنامه‌های آسیب‌پذیری که ابزار آسیب‌پذیر تشخیص نداده است.
- برنامه امن (true negative) TN: تعداد برنامه‌های امنی که ابزار به درستی در آنها آسیب‌پذیری کشف نکرده و امن تشخیص داده است.
- ستون "دیگر" نشان‌دهنده‌ی تعدادی دیگر از آسیب‌پذیری‌های کشف شده می‌باشد.
- "*" نشان‌دهنده‌ی ابزارهای غیرآکادمیک، "-" نشان‌دهنده‌ی موارد اجرانشدنی و "؟" نشان‌دهنده‌ی تعداد نامعلوم است.
- در نتیجه‌ی ارزیابی این ۱۴ ابزار، ۴ ابزار هیچ‌یک از آسیب‌پذیری‌های مطرح شده در معیار جیرا را کشف نکردند و هیچ‌کدام از ابزارها بیشتر از ۱۵ آسیب‌پذیری از ۴۲ آسیب‌پذیری را کشف نکردند. این ارقام نشان می‌دهد که هر کدام از ابزارهای جاری، بصورت جداگانه، توانایی بسیار محدودی در کشف آسیب‌پذیری‌های شناخته شده در معیار جیرا دارند. همچنین مشاهده شد که ابزارهایی که مبتنی بر تحلیل عمیق هستند تشخیص‌های درست کمتری نسبت به ابزارهای مبتنی بر تحلیل سطحی دارند و ابزارهای مبتنی بر تحلیل سطحی همه‌ی آسیب‌پذیری‌های کشف شده توسط ابزارهای مبتنی بر تحلیل عمیق را کشف کردند.

علاوه بر این در بین ابزارها، اکثر ابزارهای آکادمیک مبتنی بر تحلیل عمیق هستند در حالیکه بیشتر ابزارهای غیرآکادمیک مبتنی بر تحلیل سطحی هستند پس می‌توان نتیجه گرفت که ابزارهای غیرآکادمیک عملکرد نسبتا بهتری دارند.

بررسی‌ها نشان داد که حتی به کارگیری همه‌ی این ۱۴ ابزار با یکدیگر برای کشف آسیب‌پذیری‌های ضبط شده در جیرا، کافی نیست. ابزارها باید بتوانند بدون آسیب‌پذیر تلقی کردن برنامه‌های امن، به درستی برنامه‌های آسیب‌پذیر موجود در جیرا را آسیب‌پذیر تشخیص دهند.

از لحاظ فرمت مورد پشتیبانی توسط ابزارها، همه این ۱۴ ابزار از تحلیل فرمت apk پشتیبانی می‌کردند و از لحاظ زمان اجرا، ابزارهایی که تحلیل را به پایان می‌رساندند دارای زمان اجرای خوبی بودند.

۶ ارزیابی ابزارهای کشف رفتار مخرب

برای انجام این ارزیابی، ۶ ابزاری که در ارزیابی ابزارهای کشف آسیب‌پذیری بعنوان ابزار کشف رفتارهای مخرب حذف کردیم، مورد بررسی قرار می‌دهیم. در بین این ۶ ابزار، StaDyna را حذف کردیم چرا که این ابزار رفتار مخرب را از طریق بارگذاری کد بصورت پویا کشف می‌کرد ولی در جیرا معیاری موجود نیست که از روش بارگذاری کد بصورت پویا استفاده کند.

هیچ کدام از این ابزارها توانایی فاش کردن رفتار مخربی را که کشف می‌کرد نداشت و تقریبا هیچ‌کدام از این ابزارها بصورت محلی در دسترس نبودند. بنابراین ابزارهای کشف رفتار مخرب، محدود به توانایی کشف خود هستند (احتمالا برای مطلع نشدن برنامه‌نویسان مخرب از اینکه عملکرد مخربشان کشف شده است).

بطور خلاصه هیچ‌کدام از این ابزارها نتوانستند هیچ یک از رفتارهای مخرب حال حاضر در جیرا را کشف کنند. از آنجایی که فقط ۵ ابزار در مقابل ۳۳ اکسپلویت بررسی شدند، این ارزیابی نباید به کل ابزارها تعمیم یابد. با این حال بررسی‌ها نشان می‌دهد که احتمالا ابزارهای کشف رفتار مخرب در برنامه‌های اندرویدی موثر و کافی نیستند و ارزیابی‌های بیشتری بر روی ابزارها و اکسپلویت‌های بیشتری نیاز است.

۷ منابع

[1] Ranganath, V. P., & Mitra, J. (2018). Are Free Android App Security Analysis Tools Effective in Detecting Known Vulnerabilities?. arXiv preprint arXiv:1806.09059.

URL:https://bitbucket.org/secure-it-i/android-app-vulnerability-benchmarks/src/RekhaEval-3.

[2] Arzt, S., Rasthofer, S., Fritz, C., Bodden, E., Bartel, A., Klein, J., ... & McDaniel, P. (2014,

June). Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps. In Acm Sigplan Notices (Vol. 49, No. 6, pp. 259-269). ACM.

[3] Calzavara, S., Grishchenko, I., & Maffei, M. (2016, March). HornDroid: Practical and sound static analysis of Android applications by SMT solving. In 2016 IEEE European Symposium on Security and Privacy (EuroS&P) (pp. 47-62). IEEE.

[4] Devknox.Documentation[Online,accessed30.4.2017].Availableat URL:devknox.io/documentaion/