مجلة جامعة تشرين للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (39) العدد (3) 2017
Tishreen University Journal for Research and Scientific Studies - Engineering Sciences Series Vol. (39) No. (3) 2017

تحليل وكشف البرمجيات الخبيثة في أنظمة التشغيل للهواتف الذكية
دراسة حالة نظام التشغيل (أندرويد)

الدكتور قاسم قبلان*
أحمد عاقل**

(تاريخ الإيداع 30 / 11 / 2016. قُبِل للنشر في 17/ 5 / 2017)

□ ملخّص □

تطورت الهواتف الذكية في السنوات القليلة الماضية من مجرد هواتف محمولة بسيطة إلى كمبيوترات متطورة، وقد سمح هذا التطور لمستخدمي الهواتف الذكية بتصفح الإنترنت ،تلقي وإرسال البريد الإلكتروني، رسائل SMS و MMS والاتصال إلى الأجهزة لتبادل المعلومات. تجعل كل هذه السمات من الهاتف الذكي أداة مفيدة في حياتنا اليومية، ولكن بالوقت نفسه، تجعله أكثر عرضة لجذب التطبيقات الخبيثة.

وبمعرفة أن معظم المستخدمين يقومون بتخزين معلومات حساسة على هواتفهم المحمولة لذلك تعتبر الهواتف الذكية هدفاً مرغوباً للمهاجمين ولمطوري البرمجيات الخبيثة مما يجعل من المحافظة على أمن البيانات وسريتها على منصة أندرويد (من تحليل البرمجية الخبيثة على هذه المنصة) قضية ملحّة.

اعتمد هذا البحث على أساليب التحليل الديناميكي لسلوك التطبيقات حيث تبنى أسلوب لكشف البرمجيات الخبيثة على منصة أندرويد. تم تضمين برنامج الكشف في إطار تجميع آثار عدد من المستخدمين واعتمد على برنامج crowdsourcing حيث تم الاختبار بتحليل البيانات المجمعة عند المخدم المركزي باستخدام نوعين من البيانات: البيانات التي تم الحصول عليها من البرمجيات الخبيثة الصنعية لأغراض الاختبار وأخرى من البرمجيات الخبيثة الواقعية. تبين أن الأسلوب المستخدم وسيلة فعالة لعزل البرمجيات الخبيثة وتنبيه المستخدمين بها .

الكلمات المفتاحية: أمن نظام الأندرويد ، البرمجيات الخبيثة،سوق غوغل ، مستودعات أندرويد.

* مدرس – قسم النظم والشبكات الحاسوبية – كلية الهندسة المعلوماتية – جامعة تشرين – اللاذقية – سورية.
** طالب دراسات عليا (ماجستير)– قسم النظم والشبكات الحاسوبية – كلية الهندسة المعلوماتية – جامعة تشرين– اللاذقية – سورية.

مجلة جامعة تشرين للبحوث والدراسات العلمية _ سلسلة العلوم الهندسية المجلد (39) العدد (3) 2017
Tishreen University Journal for Research and Scientific Studies - Engineering Sciences Series Vol. (39) No. (3) 2017

Analysis and detect malicious software in the operating systems for smart phones Operating System Case Study (Android)

Dr. Kassem Kablan [*]
Ahmad Akel [**]

(Received 21 / 6 / 2016. Accepted 21 / 1 / 2017)

□ ABSTRACT □

In the past few years, smart phones developments than just a simple mobile phones to sophisticated computers. This development has allowed for users of smart phones to surf the Internet, receive and send e-mail, SMS and MMS messages and connect to devices to exchange information. And make all these features of the smartphone useful tool in our daily lives, but the same time, make it more useful to attract malicious applications. Knowing that most users store sensitive information on their mobile phones .smart phones are considered desirable scorer attackers and malware developers .And make the need to preserve the security and confidentiality of the data on the Android platform from malware analysis on it is an urgent issue.

This research on previous methods to analyze the dynamic behavior of the applications have been approved and adopted a method to detect malware on the Android platform. It was implication the reagent in the context of assembling the effects of the number of users relied on crowdsourcing. We have been testing our frame analysis of the collected data at the central server using two types of data sets: data from the artificial malware have been created for testing purposes and malware incident of life in the world. It turns out that the method used is an effective way to isolate malware and alert users to software that was downloaded.

Key words: Android Security, Malware ,Google Market ,Android Store.

[*]**Assistant Professor, Department of System and Computing Network Engineering , Faculty of Informatics, Engineering, Tishreen University, Lattakia, Syria.**
****Postgraduate Student, Department of System and Computing Network Engineering , Faculty of Informatics, Engineering, Tishreen University, Lattakia, Syria.**

مقدمة:

لقد هددت البرمجيات الخبيثة الكمبيوترات لسنوات ونتيجةً للنمو السريع لمبيعات الهواتف الذكية، كانت القضية هنا هي السؤال عن الزمن الذي قد يصبح خلاله مطورو البرمجيات الخبيثة مهتمون بمنصات الهواتف الذكية للقيام بالهجمات . ووفقاً لدراسة اجرتها شركة البيانات الدولية International Data Corporation، سيشحن مصنعو الهواتف الذكية(أكثر من 850 مليون جهاز في 2016)، بالمقارنة مع 503.4 مليون وحدة تم شحنها في 2015 [1]. أيضاً سينمو سوق الهواتف الذكية بشكل أسرع بأربعة أضعاف من سوق الهواتف المحمولة وسيرتفع الطلب على الهواتف الذكية باستمرار ويصل إلى الحد حيث يستبدل العملاء هواتفهم المحمولة القديمة بهواتف ذكية.

لقد أدى نمو مبيعات شركات الهواتف المحمولة مثل Samsung وHTC بين 2013 و 2015 إلى خلق ثورة في سوق الهواتف الذكية. ووفقاً لهذا، توقعت IDC(Intrusion Detection System) أن نظام التشغيل أندرويد سيقود سوق أنظمة تشغيل الهواتف الذكية في السنوات القادمة. وأيضاً، أن نظام التشغيل أندرويد قد ازداد بحدود 50% بين 2012– 2016، مع احتمالية مرتفعة لأن تصبح من الموزعين الرائدين لأنظمة تشغيل الهواتف المحمولة في المستقبل [3].

يعتبر سوق غوغل هو الآلية الرسمية لنقل البرمجيات إلى هاتف ذكي معتمد على الأندرويد. ولسوء الحظ يمكن لمطوري تطبيقات الأندرويد تحميل التطبيقات إلى الإنترنت بدون أي تحقق بخصوص مصداقيتهم. وتكون التطبيقات موقعة من قبل المطورين أنفسهم. وتوجد مستودعات غير رسمية، حيث يمكن للمطورين تحميل التطبيقات، بما في ذلك التطبيقات المفكوك تشفيرها cracked أو Trojans. وقد سمح هذا للمهاجمين الخبيثين القيام بتحميل البرمجيات الخبيثة إلى السوق Market وأيضاً لنشر البرمجيات الخبيثة عبر مستودعات غير رسمية.

إن مركز شركة Juniper Networks للتهديد العالمي Global Threat Center لاحظ زيادة 400% في برمجيات أندرويد الخبيثة منذ صيف 2013. وتعتبر "Fake Player", "Geinimi", "PJApps", "HongToutou" هي بعض الأمثلة المعروفة عن البرمجيات الخبيثة . وقد تم تحديد عدد من التطبيقات وربط البرمجيات الخبيثة، ونشرها عبر المستودعات(Stores) غير الرسمية. حيث تم إيجاد أكثر من 50 تطبيق مصاب بالفيروسات في آذار 2014، وكلها مصابه بتطبيقات Trojan "DroidDream"، ومؤخراً أثبت John Oberheide مفهوم تطبيقات البرمجيات الخبيثة مثل مكافأة Angry Birds لإظهار ضعف أمن سوق الأندرويد Android Marketplace.

في هذا البحث سنقترح مبدأ جديد لتحليل سلوك تطبيقات أندرويد، من حيث تأمين إطار للتمييز بين التطبيقات التي تتصرف بشكل مختلف، مع أن لها الاسم نفسه والنسخة نفسها. ويكون الهدف هو كشف التطبيقات التي تتصرف بشكل مشكوك به، وهكذا تكشف البرمجيات الخبيثة في شكل Tojans (فيروس حصان طروادة).

أهمية البحث وأهدافه:

المساهمة الرئيسية لهذا العمل هي استخدام نظام crowdsourcing للحصول على آثار لتطبيقات السلوك، والتي تساعد الباحثين في تجميع عينات مختلفة من آثار تنفيذ التطبيقات. ويمكن تقسيم هذه الآثار إلى مجموعتين مختلفتين، مما يؤدي إلى تمييز واضح بين التطبيقات الحميدة و تلك المحتوية على البرمجيات الخبيثة.

أن النظام المقترح سيكون قادر على كشف تنفيذ كل البرمجيات الخبيثة في اختبارات البرمجيات الخبيثة المكتوبة ذاتياً، مما يعطي معدل كشف 100% من أجل البرمجيات الخبيثة. وتؤمّن تحليل وكشف حقيقي للبرمجيات الخبيثة المتواجدة في سوق غوغل.

طرائق البحث ومواده:

تم تنظيم هذا العمل كما يلي. يصف القسم 4 الأعمال السابقة. وفي القسم 5 نشرح إطار نظام كشف البرمجيات الخبيثة المعتمد على السلوك، تفصيل عملية بناء تطبيق crowdsourcing للتجميع وإعطاء المعلومات بخصوص نظام كشف البرمجيات الخبيثة المنفذ مع مجموعة من تطبيقات البرمجيات الخبيثة المكتوبة ذاتياً المحتوية على البرمجيات الخبيثة. وفي القسم 6 نختتم ونقدم العمل المستقبلي الممكن لتقييل التقييدات للنظام المقترح.

وقد تم استخدام محاكي الأندرويد(Android Emulator) لتجربة التطبيقات كما تم استخدام برنامج MATLAB لنمذجة نتائج التجربة.

الأعمال السابقة

حتى الآن تم اقتراح أسلوبين من أجل تحليل وكشف البرمجيات الخبيثة: التحليل الستاتيكي والتحليل الديناميكي . يعتمد التحليل الستاتيكي، المستخدم غالباً من قبل شركات مضادات الفيروسات، على كود المصدر أو الفحص على مستوى البت الذي يبحث عن أنماط مشكوك بها. ومع أن بعض المبادئ كانت ناجحة، فقد طور مؤلفو البرمجيات الخبيثة تقنيات غش متنوعة التي قد تكون فعالة خصوصاً ضد التحليل الستاتيكي. ومن جهة أخرى، يشمل التحليل الديناميكي أو الكشف المعتمد على السلوك تشغيل العينة في بيئة مستقلة ومعزولة من أجل تحليل آثار تنفيذها. ويؤمن Egele مراجعة شاملة لتقنيات تحليل البرمجيات الخبيثة الديناميكية المؤتمتة.

بدّل .David Dagon [1] المجتمع في 2011 بتوقع جدوى البرمجيات الخبيثة في الهواتف المحمولة. وحتى wi-fi والبلوتوث تم اعتبارها على أنها مسارات العدوى الأكثر احتمالاً، نمو مبيعات الهواتف الذكية جعلت التوقع يصبح أكثر مصداقية. وبشكل أساسي، في حزيران من السنة نفسها، تم اكتشاف البرمجيات الخبيثة الأول المكتوب بشكل خاص من أجل منصة نظام تشغيل اندرويد. وبعد نجاح العدوى الذي قام به البرمجيات الخبيثة الـ Cabir وتنويعاته، اقترح الباحثون مبادئ أساسية وطوروا آليات مختلفة من أجل كشف البرمجيات الخبيثة في الهواتف الذكية.

ونتيجةً لنقص أنماط البرمجيات الخبيثة للهواتف الذكية في ذلك الوقت، استخدمت معظم تقنيات كشف الشذوذ مبدأ استهلاك طاقة البطارية كسمة نظام كشف رئيسي للبرمجيات الخبيثة. واعتمدت هذه التقنيات على التحقق من استهلاك طاقة بطارية الهواتف المحمولة ومراقبته ومقارنته مع نمط استهلاك الطاقة العادي لكشف الشذوذات. وتم تصميم هذه التقنيات بشكل خاص من أجل كشف الهجمات التي تستهدف عمر البطارية.

لقد أدت تقييدات التجهيزات للهواتف الذكية قيام الباحثين باقتراح تقنيات تحليل جمعية، حيث تم التحليل بواسطة شبكة من الأجهزة. وقد تم اقتراح كل من التحليل الستاتيكي والديناميكي باستخدام هذه التقنيات.

كما تم اقتراح أعمال التحليل الستاتيكي من أجل كشف البرمجيات الخبيثة في الهواتف الذكية الفردية. وقد كيفت شركات مضادات الفيروسات أنظمة كشفها المعتمدة على التوقيع من أجل الهواتف الذكية، ولكن تعتبر مستوى المصادر المطلوبة من قبل تقنيات مضادات الفيروسات وتقييدات الطاقة والذاكرة للأجهزة المحمولة، لم يكن التحليل ضمن الهاتف حلاً مفضلاً ليتم تطبيقه إلى الهواتف الذكية. وقد اقترح Schmidt تحليل طلبات وظيفية استاتيكية

من الثنائيات التي تطبق خوارزمية تجميع. وتم استخدام هذه التقنية لكشف البرمجيات الخبيثة لنظام التشغيل لمتطلبات الهواتف المحمولة، مثل كفاءة الجهاز، سرعة استخدام المصادر ومحدوديته. وقد تم أيضاً اقتراح كشف الشذوذ لكشف البرمجيات الخبيثة في أجهزة اندرويد.

بينت شركات مضادات الفيروسات أنه يمكن نشر التحليل الستاتيكي من أجل كشف البرمجيات الخبيثة في أجهزة أندرويد. ولكن ونتيجةً للمصادر المحدودة للهواتف الذكية، اعتمدت معظم الاقتراحات الحديثة من أجل كشف البرمجيات الخبيثة على أجهزة أندرويد على تحليل السلوك من أجل كشف الشذوذ.

اقترح .Schmidt [2] حلاً بالاعتماد على حوادث المراقبة الحاصلة على مستوى نواة استدعاء النظام (Linux kernel). ومراجعة الأدوات المعتمدة على linux من أجل تحسين الأمن، واستخلاص السمات مثل طلبات النظام، الملفات المعدلة، إلخ. من نواة linux. ومن ثم تم استخدام هذه السمات لخلق نموذج عادي من أجل سلوك الهاتف الذكي.

عرض Enck TaintDroid في [3]. واستخدم نظامها تحليل ملفات ال batch الديناميكي لمراقبة معلومات حساسة في الهواتف الذكية. وهكذا، يمكن أن تتعقب معلومات موضع أو تحديد معلومات كتاب. والتطبيق الذي يستخدم البيانات الحساسة لا يتوافق بالضرورة إلى البرمجيات الخبيثة.

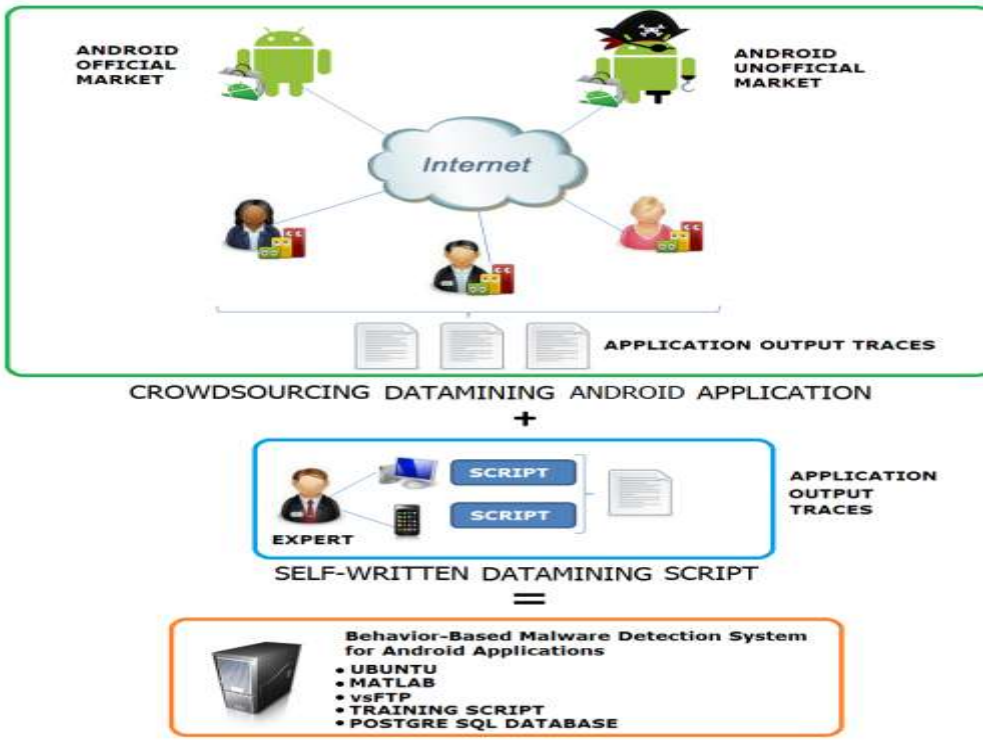
اقترح Portolakidis في Paranoid Android، نظاماً حيث يمكن للباحثين إجراء تحليل البرمجيات الخبيثة كامل في السحابة باستخدام نسخ الهاتف المحمول. ويتوجب تشغيل مبدأهم لتلك النسخ في بيئة افتراضية آمنة، تم تقييد نظامهم إلى أكثر من 105 نسخ تعمل بالتزامن. ومن ثم يمكن تطبيق تقنيات كشف البرمجيات الخبيثة مختلفة.

أخيراً، عرض .Shabtai في [5] منهجية لكشف الأنماط الزمنية المشكوك بها مثل السلوك الخبيث المعروف باسم التجريد الزمني المعتمد على المعرفة knowledge-based temporal abstraction.

1 – إطار نظام كشف البرمجيات الخبيثة المعتمد على السلوك

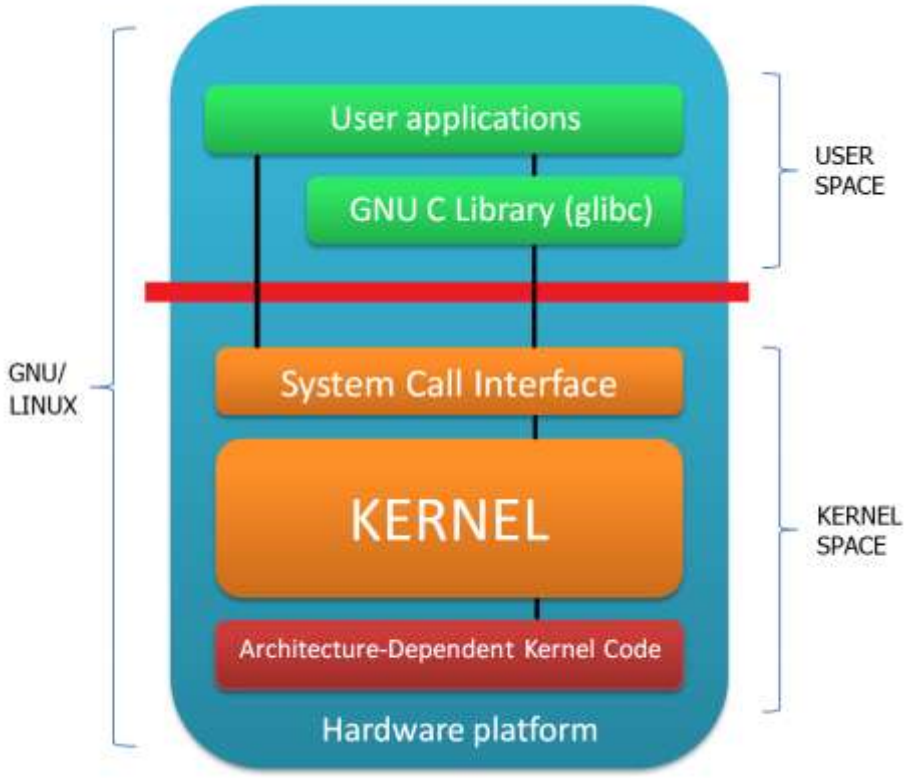
إن تطبيق أنظمة كشف البرمجيات الخبيثة في الأجهزة المحمولة مفهوم جديد نسبياً. والأدوات الأمنية والآليات المستخدمة في الكمبيوترات ليست مجدية من أجل تطبيق الهواتف المحمولة نتيجةً لاستهلاك المصادر الزائدة واستنزاف البطارية. ومن ثم، قررنا القيام بعملية تحليل كامل على مخدم بعيد. وسيتم استخدام هذا المخدم حصرياً لتجميع المعلومات ولكشف التطبيقات الخبيثة المشكوك بها في منصة أندرويد.

يتركب إطارنا من عدة مكونات تؤمن مصادر وآليات كافية لكشف البرمجيات الخبيثة على منصة أندرويد. أولاً، قمنا بتطوير برنامج بسيط يسمى Crowdroid، والذي يمكن تنزيله وتنصيبه من سوق غوغل. ويكون هذا التطبيق مسؤولاً عن مراقبة طلبات استدعاء نظام نواة Linux Kernel linux وإرسالها بعد معالجتها مسبقاً إلى مخدم مركزي. ووفقاً لفلسفة crowdsourcing، سيساعد المستخدمون بإرسال بيانات غير شخصية ولكن متعلقة بالسلوك لكل تطبيق يستخدمونه. وربما يتم تنزيل هذه التطبيقات من السوق الرسمية Market وأيضاً من stores غير الرسمية كما هو موضح في الشكل 1 [4].



الشكل(1.) كشف البرمجيات الخبيثة على منصة أندرويد

ومن ثم، سيكون المخدم البعيد مسؤولاً عن فرز البيانات، وانشاء مصفوفة متجهات لطلبات استدعاءات النظام لكل تفاعل للمستخدمين ضمن تطبيقاتهم. وهكذا سيتم خلق مجموعة بيانات السلوك لكل تطبيق مستخدم. وكلما زاد عدد المستخدمين الذين يستخدمون تطبيق Crowdroid، سيكون النظام أكثر اكتمالاً ودقةً. أخيراً، نقوم بتجميع كل مجموعة بيانات باستخدام خوارزمية التجميع المجزأة. وبهذه الطريقة يمكننا التمييز بين تطبيقات جيدة تظهر أنماط طلب استدعاء للنظام مشابهة جداً، وتطبيقات Trojan خبيثة والتي، حتى لو كان لها الاسم والمحدد نفسه، لها سلوك مختلف وفقاً للمسافة بين متجهات التطبيقات المتشكلة. يكون التجميع الجزئي هو ببساطة تقسيم Clusters (التجميعات) بحيث أن كل جسم بيانات يكون تماماً مجموعة جزئية واحدة. وربما يتم تمثيل كل تجميع بواسطة مركز متوسط أو تحميع تمثيلي. وتحاول خوارزميات التجزئة إما اكتشاف التجميعات بنقاط إعادة التموضع التكرارية بين المجموعات الجزئية (التجميع الاحتمالي، أساليب medoids، أساليب متوسطات k)، أو محاولة تحديد التجميعات على أنها مناطق ذات كثافة جيدة بالبيانات (تجميع معتمد على الكثافة). اخترنا خوارزمية متوسطات k [4] نتيجةً لبساطتها، كفائتها، سرعتها، والعدد المعروف البالغ k = 2 للتجميعات كمعيار إدخال: نعرف أن التطبيق سيكون جيداً أو خبيثاً. يكون في linux، استدعاء نظام linux هو كيف يطلب برنامج خدمة من نواة نظام تشغيل. يكون لـ Linux kernel 2.6.23 أكثر من 250 استدعاء نظام ويتم تحديد كل واحد بعدد فريد يكتب في جدول استدعاء النواة. تؤمن استدعاءات النظام وظائف مفيدة لبرامج التطبيق مثل عمليات تشغيل شبكة، ملف، أو عملية متعلقة بالتشغيل. وكما هو موضح في الشكل 2 [4].



الشكل(2.) مستخدم linux وفضاء النواة

عندما يعمل تطبيق في فضاء مستخدم طلباً إلى نظام تشغيل Operating System، يتوجه الطلب عبر مكتبة glibc(GNU Library C)، واجهة طلب النظام System Call Interface، و Kernel وأخيراً إلى العتاد Hardware، تعالج مكتبة glibc الطلب ويوجه CPU النواة لتنفيذ وظيفة نواة مناسبة للانتباه لجدول طلب نظامك. وستكون النواة مسؤولة عن فهم الطلب والقيام بطلب إلى منصة العتاد. وبعد ذلك يحصل المستخدم على المعلومات المطلوبة من قبل التطبيق في فضاء المستخدم في عملية معكوسة. وتكون الوظائف مثل:

()socket(طريقة لتأسيس اتصال)

()read(طريقة للقراءة)

()open(طريقة لفتح اتصال)

()getpid(طريقة لإعطاء معرف)

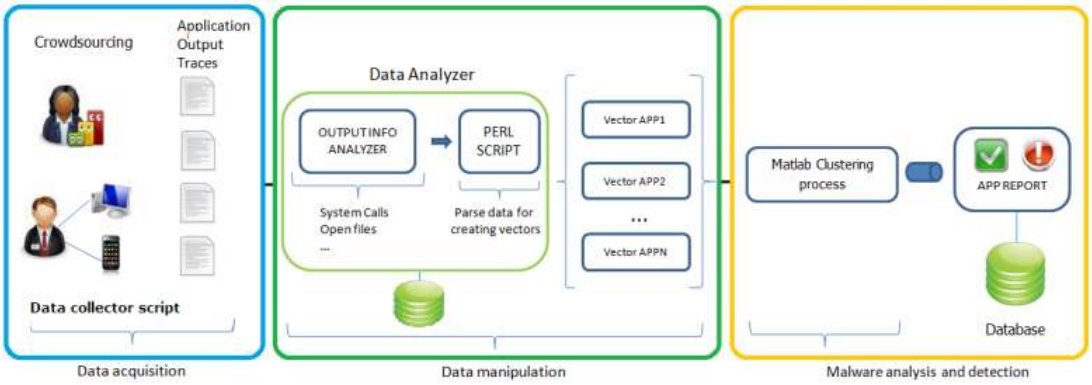
هي بعض الوظائف التي تستطيع glibc تأمينها للتطبيقات لتحريض طلب النظام. ويتم تنفيذ نواة linux في الطبقة الدنيا من بنية أندرويد. وهذا يعني أن كل الطلبات التي تمت من الطبقات العليا تمر عبر النواة باستخدام واجهة طلب بيني قبل أن يتم تنفيذها في الهاردوير. وسيؤمن التقاط وتحليل طلبات النظام التي تمر عبر واجهة طلب النظام، معلومات دقيقة بخصوص سلوك التطبيق. وسيستخدم Crowdroid أداة متوافرة في linux تسمى Strace لتجميع طلبات النظام. ويكون هدف هذه الأداة استدعاء طلبات النظام لتوليد ملف خرج مع بعض الحوادث المتولدة بواسطة تطبيق أندرويد. وسيؤمّن هذا الملف معلومات مفيدة، مثل الملفات المفتوحة والتي تم الوصول إليها، أختام زمن التنفيذ وتعداد رقم كل استدعاء نظام منفذ بالتطبيق. وسنستخدم هذه السمة الأخيرة لتمثيل السلوك في كل تنفيذ تطبيق أندرويد.

بعد ذلك نعرض متجه استدعاء نظام سلوك تطبيق أندرويد ويمثل كل عنصر تعداد طلب النظام المحدد المطلوب. مع العلم أن المجموعة المكتملة لاستدعاءات نظام أندرويد كبيراً جداً ليتم عرضه في هذاالعمل، ولكن قد يجدها القارئ في صفحات دليل نواة linux.

0, 1649 , 0, 0, 0, 0, 0, 9, 260, 0, 0, 0, 0, 0, 0, 12, 0, 0, 0, 0, 0, 0, 34, 4, 47, 25, 0, 0, 0
0, 0, 0, 3466, 0, 0, 0 , 0, 0, 0, 0, 0, 0, 22, 0, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 10, 0, 0, 0, 0, 0,
0, 0, 0, 41, 40, 0, 0, 0, 0, 0, 0, 0, 0, 132 , 0, 0, 0, 0, 12, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
...,0 ,17 ,87 ,0 ,4 ,0 ,0 ,00 ,0 ,0 ,76 ,0 ,0

نمثل الأرقام(المفصولة بفواصل)، عدد الطلبات/مرات التنفيذ تم عمله بواسطة تطبيق أندرويد نوعي خلال عملية المراقبة. ومثلاً، يستخدم طلب النظام ()open 25 مرة، و()kill 47 مرة. هذا يعني أن التطبيق المراقب يفتح الملفات أو مكتبات النظام(استدعاء نظام) 25 مرة وتكون العمليات المقتولة هي 47 مرة، وهكذا.

ينشأ الإطار العديد من مجموعات البيانات كمحددات تطبيق، لذا نحتاج للعديد من مستخدمي (Crowdroid) قدر الإمكان لإغناء قاعدة البيانات ، تأمين معلومات كافية تشمل كافة تأثيرات التطبيقات الحميد ة والخبيثة حتى يستطيع النظام اكتشاف الشذوذات. ويظهر الشكل 3 بنية كامل الإطار .



الشكل(3) العملية الكاملة لكشف البرمجيات الخبيثة لأندرويد

وبالتلخيص، يعتمد الإطار على مكونان اساسيان: اكتساب البيانات ومعالجتها ، وتحليل البرمجيات الخبيثة ونظام الكشف.

- اكتساب البيانات ومعالجتها: يكون هذا المكون مسوؤلاً عن الحصول على بيانات تطبيق من المستخدمين، باستخدام تطبيق Crowdroid. تتركب البيانات المتجمعة بواسطة معلومات الجهاز الأساسية، لائحة التطبيقات المركبة، ونتيجة تطبيقات المراقبة مع ملف سجل استدعاءت النظام لأداة Strace . يتم تخزين معلومات النظام الأساسية في قاعدة بيانات مركزية، وتمت معالجة آثار استدعاء النظام لإنتاج متجهات السمة التي سيتم استخدامها من أجل التجميع.
- تحليل البرمجيات الخبيثة وكشفه: يكون هذا المكون مسؤولاً عن تحليل وتجميع متجهات التطبيقات التي تم الحصول عليها من الطور السابق من أجل خلق نموذج الاعتياد وكشف السلوك الغريب في تطبيقات أندرويد، باستخدام تجميع متوسطات K عبر متجهات استدعاء طلب النظام.

النتائج والمناقشة

في هذا القسم نعرض النتائج التفصيلية للتجارب المنفذة باستخدام الاسلوب المقترح. نختبر نظامنا مع نوعين مختلفين من البرمجيات الخبيثة. الأول، البرمجيات الخبيثة المكتوبة ذاتياً من قبل المستخدم يعطينا 100% معدل كشف. ومن أجل خلق نموذج اعتيادي للبرامج المكتوبة ذاتياً، تم استخدام آلة حاسبة، تطبيق العد التنازلي، ومحول النقود (Calculator G، Countdown G، eyConverter G –Mon). بحيث تم تطوير نسخ معدلة لهذه التطبيقات من أجل محاكاة Trojan الخبيث.

بعد ذلك اختبرنا كامل الإطار باستخدام عينتين حقيقيتين من البرمجيات الخبيثة: PJApps المحتوى في تطبيق ويندوز Steamy، و HongTouTou وهو Trojan مقترن في تطبيق Monkey Jump 2. وقد استخدمنا خدمة الذكاء الخبيث الفيروسي Virustotal البرمجيات الخبيثة Intelligence Service للحصول على التطبيقات المصابة، قد يتم تحميل التطبيقات الحميدة من الانترنت يدوياً. في هذه الحالة، حصلنا على دقة كشف 100% من أجلPJApps و85% من أجل HongToutoutrojan.

يمكن تلخيص الإعداد من أجل التجارب كما يلي:

- استخدمنا 20 زبون يشغل تطبيق Crowdroid. حيث أنه في المستقبل سيساهم المزيد من المستخدمين في النظام، وهكذا ستكون مجموعة البيانات أغنى في نظامنا.
- سنعتبر 60 تبادل من المستخدمين مع كل تطبيق كاف من أجل اكتشاف البرمجيات الخبيثة في تطبيقات مكتشفة ذاتياً: بينما يكون قد تم اختبار البرمجيات الخبيثة مع نمطنا الأولي باستخدام 6 و 20 تفاعلاً. وقد كان هذا نتيجةً لنقل الحشد المتوافر عند وقت تجاربنا.
- تحويل البيانات/ تواصلها بين تطبيق العملي والسيرفر يستخدم بروتوكول FTP (file transfer protocol).
- سيخلق نظامنا نماذج للتمييز بين التطبيقات الحميدة والخبيثة، مثل Trojans. ولن يتم اكتشاف البرمجيات الخبيثة غير معروف بدون برمجيات جيدة متعلقة.
- ونفترض أن التطبيقات الحميدة هي تلك التي تم تنفيذها مرات أكثر. وسيتم تحميل Trojans إلى مستودع غير رسمي لاحقاً أكثر من التطبيق الأصلي، طالما أن كاتب البرمجيات الخبيثة سيكون بحاجة للتطبيقات الأصلية من أجل خلق برامج كهذه.

1. البرمجيات الخبيثة المكتوبة ذاتياً (الصنعية)

تم عرض نتائج نظام كشف البرمجيات الخبيثة أندرويد المعتمد على السلوك على تطبيقات مكتوبة ذاتية في الجدول 1 ولاختبار النظام، حصلنا على 60 أثر تنفيذ من كل من التطبيقات المكتوبة ذاتياً، كان 50 أثر منها هو تطبيقات حميدة 10 آثار هي خبيثة. وستمثل التفاعلات 50 الحميدة نموذج التسوية للتطبيق. يجمع النظام كل ملفات الخرج المتولدة من كل تفاعل ويخلق 3 ملفات، واحد لكل تطبيق. ونحصل أخيراً على 60 متجه استدعاء واحد لكل تطبيق (الآلة الحاسبة، العد التنازلي، ومحول النقود) بما في ذلك متجهات استدعاء سلوك تطبيق البرمجيات الجيدة و البرمجيات الخبيثة.

(الجدول 1 نتيجة التطبيقات الخبيثة لأندرويد المكتوبة ذاتياً)

	Interactions		Clustering result		Detection rate
	Good	Malware	Good Clustered	Malware Clustered	
Calculator	50	10	50	10	100%
Countdown	50	10	50	10	100%
MoneyConverter	50	10	50	10	100%

وكما هو موضح في الجدول أعلاه، سيكون النظام قادراً على تصنيف متجهات الاستدعاءات بتجميعين مختلفين، تجميع تفاعلات التطبيقات الحميدة والخبيثة بشكل صحيح.

2. البرمجيات الخبيثة الحقيقية

طالما أن النتائج التي تم الحصول عليها من البرمجيات الخبيثة المكتوبة ذاتياً في النظام كانت ناجحة، قررنا عمل تحليل أعمق من أجل البرمجيات الخبيثة المحتو اة في تطبيقات Steamy Window و 2 Monkey Jump ، باستخدام عميل Crowdroid.

1.2. تطبيق Steamy Window مع البرمجيات الخبيثة

Steamy Window هو تطبيق حر عند سوق الأندرود الذي يغطي شاشة الهاتف الذكي بالبخار ويسمح للمستخدم بمسحه بأصابعه. ترسل النسخة الخبيثة من التطبيق المحتوية على PJApps البرمجيات الخبيثة، والتي تم اكتشافها في المخازين غير الرسمية، معلومات حساسة تحتوي على IMEI، Device ID، Line Number و Subscriber ID إلى سيرفر الويب. ومن ثم يصبح الهاتف الذكي مسجلاً في Command and Control botnet بانتظار التعليمات. وتكون له المقدرة على إرسال رسائل نصية إلى أعداد معدل من الدرجة الأولى، SMS-spamming، تنصيب المزيد من التطبيقات، الملاحة، وحتى حجز مواقع الويب.

من أجل Steamy Window، تم القيام بستة تفاعلات لاختبار النظام، 4 تستخدم تطبيق Steamy Window الأصلي و 2 كود خبيث لـ PJApps البرمجيات الخبيثة متصل. ومتجهات استدعاء طلب النظام حيث تجمعت وتراكبت مع خوارزمية متوسطات k. ويظهر الشكل 4 كيف أن النظام حصل على التطبيقات من مصادر مختلفة. وينصب بعض المستخدمين تطبيق Official Steamy Window لأندرويد وآخرين يقومون بتنزيل تطبيق من مستودعات أندرويد غير رسمية.

يظهر الجدول 2 مصفوفة المسافة بين كل تفاعل لتطبيقات Steamy Window مع مسافة إقليدية كمصفوفة تشابه:

(الجدول2 مصفوفة مقارنة متجهات استدعاء نظام Steamy Window)

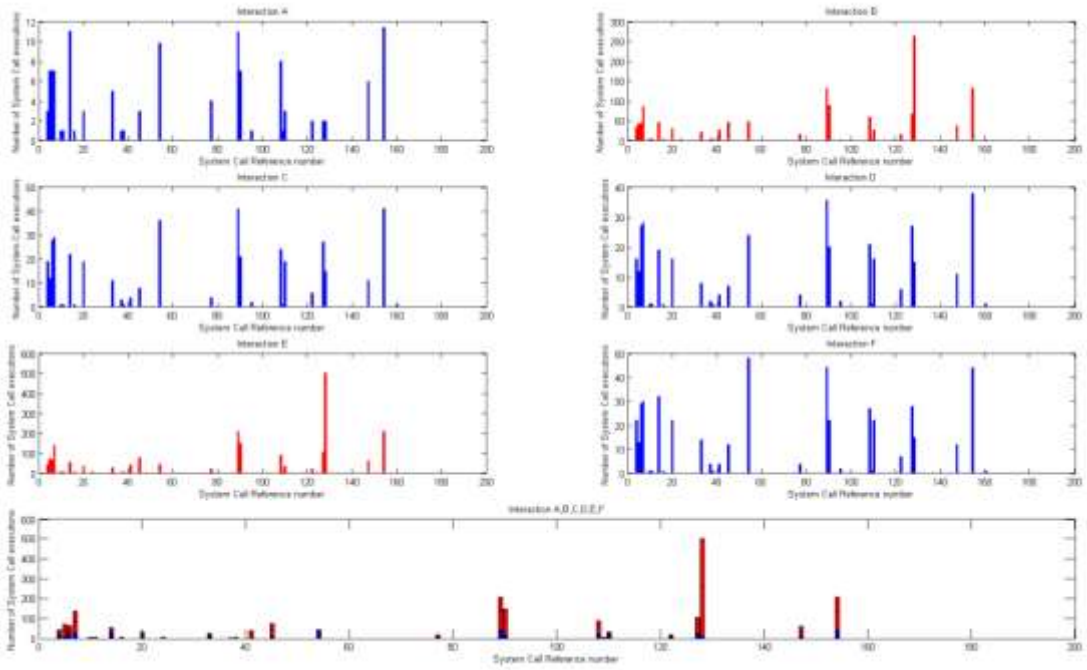
Interaction	A	B	C	D	E	F
A	0	0.1818	0.1414	0.1414	0.1818	0.1414
B	0.1818	0	0.1768	0.1768	0.1616	0.1667
C	0.1414	0.1768	0	0.1010	0.1818	0.1212
D	0.1414	0.1768	0.1010	0	0.1818	0.1212
E	0.1818	0.1616	0.1818	0.1818	0	0.1717
F	0.1414	0.1667	0.1212	0.1212	0.1717	0

المسافات القريبة إلى 0، هي متجهات مماثلة أو مشابهة. المسافات ذات قيمة أكبر من 0، تكون متجهات غير متماثلة. وتظهر مصفوفة المسافة أن قيم التفاعلات B و E بالمقارنة إلى تفاعلات التطبيق الحميد هي أعلى من أخرى.أخيراً، تم استخدام خوارزمية متوسطات k من أجل تجميع التفاعلات. تم عرض النتائج في الجدول 3. يظهر الصف الأول رقم التجميع الذي أُعطي لكل تفاعل كنتيجةً لتطبيق متوسطات k. ويحتوي هذا الصف على رقم التجميع الذي تنتمي إليه البيانات. يظهر الصف الثاني أية تفاعلات حميدة (X) و البرمجيات الخبيثة (√). يكون النظام قادراً على أن يحدد بشكل صحيح التطبيقين الخبيثين B و E، والذي هو دليل أن نظام كشف البرمجيات الخبيثة أندرويد المعتمد على السلوك قادر على كشف عمليات التنفيذ الخبيثة لتطبيق Steamy Window.

(الجدول 3: نتيجة تجميع Steamy Window)

Interaction	A	B	C	D	E	F
Cluster	1	2	1	1	2	1
Application	✓	✗	✓	✓	✗	✓

طريقة أخرى لتمثيل متجهات طلب النظام التي تم الحصول عليها هي استخدام المخططات البيانية الشريطية كما هو موضح في الشكل 4.



(الشكل 4. المخطط الشريطي لتفاعلات Steamy Window)

شرح المخطط وتحليل النتائج

تمثل القضبان الموجودة بالمخططات (2,3) من القسم A و القضبان الموجودة بالمخططات من القسم (1,2) سلوك عمليات تنفيذ تطبيق Steamy Window الحميد، و تمثل القضبان الموجودة بالمخططات (1) من القسم A و القضبان الموجودة بالمخططات من القسم (3) سلوك عمليات تنفيذ تطبيق Steamy Window المصابة بـ PJAppstrojan. يكون لكل طلب نظام رقمه الخاص، ويمثل المحور X الرقم المخصص من أجل كل استدعاء نظام منفذ. ومن جهة أخرى، يمثل المحور Y عدد مرات التي تم بها تنفيذ استدعاء نظام. وقد قمنا بإزالة من المخطط استدعاءات النظام ذات نسبة الاستدعاء المرتفعة جداً في كل التفاعلات بما في ذلك البرمجيات الخبيثة (ioctls time, recv and ptrace) من أجل التركيز على استدعاءات النظام ذات العلاقة. يمكننا أن نرى أن النسخة المصابة بال Trojan تنفذ استدعاءات نظام أكثر وإضافية في الجهاز. وبالأخذ بالاعتبار كل من التطبيقين له النسخة نفسها، يمكننا الافتراض أن تطبيق Steamy Window المنزّل من مستودعات غير رسمية، المنفذ في التفاعلات B وE، هو تطبيقات أندرويد غير سوية. ويظهر المخطط الأخير مزيجاً من بقية القضبان. ويمكننا أن نحدد بسهولة أي استدعاء نظام يكون مسؤولاً عن سلوك كهذا. وتحديداً، استدعاءات النظام , read(), open(), access(), chmod() chown() قد كانت الأكثر علاقة. ويستخدم التطبيق الأصلي أيضاً أول استدعائين، ولكن مع تكرار أقل. ويتم تحريض access(), chmod() and chown() بواسطة البرمجيات الخبيثة، مما يسمح بالوصول وتغيير السماحيات والملكية لمجموعة من الملفات والسواقات.

الاستنتاجات والتوصيات

تتوقع كل تطبيقات سوق الهواتف الذكية زيادة كبيرة في عدد الهواتف الذكية المباعة في السنوات الخمس الآتية. مما سيخلق هنا إمكانية من أجل زيادة كبيرة في توليد البرمجيات الخبيثة، وبشكل خاص في القطاع الذي يسيطر عليه القائد السوقي، وبشكل محتمل منصة أندرويد.

لقد اقترحنا في هذا البحث إطاراً جديداً للحصول على نشاط تطبيق هاتف ذكي وتحليله. وبالتعاون مع عينة من مجتمع مستخدمي أندرويد سيكون قادراً على التمييز بين التطبيقات الحميدة والخبيثة ذات الاسم والنسخة نفسهما، و كشف السلوك الشاذ لتطبيقات معروفة. و بالإضافة الى استخدام منصتنا ورقم الهواتف الذكية الاختبارية، خلقنا إثباتاً للمفهوم لهذه الآلية، وكوسيلة لتحليل التهديدات الناشئة.

الخطوة الآتية هي نشر برنامج بسيط Crowdroid في سوق غوغل Google's Market وتوزيعه إلى أكثر عدد ممكن من المستخدمين. وسيكون للمستخدمين الذين يشغلون التطبيق الفرصة لرؤية سلوك هواتفهم الذكية. كما يمكننا تنبيه المستخدمين عندما يظهر واحد من تطبيقاتهم تعقباً غير سوي. ويمكن للنظام أن يعمل كنظام تحذير مبكر بحيث يمللك القدرة كشف التطبيقات التي تتصرف بشكل خبيث أو غير سوي في المراحل المبكرة من الانتشار.

أخيراً، المطلوب تحديداً إقناع مجتمع مستخدمي أندرويد بتنصيف تطبيق Crowdroid. ويجب علينا إدارة إدراك ضياع الخصوصية عند دعم مجتمع الأبحاث مع معلومات سلوكها، مقابل الفائدة للوصول إلى الإحصائيات الأحدث البرمجيات الخبيثة المكتشفة المعتمدة على السلوكيات.

المراجع:

[1] *50 Malware applications found on Android Official Market*. http://m.guardian.co.uk/technology/blog/2011/mar/02/android-market-apps-malware?cat=technology&type=article.

[2] GUANGDONG BAI, LIANG GU, TAO FENG, YAO GUO, and XIANGQUN CHEN. *Context-aware usage control for android*. In *Security and Privacy in Communication Networks*, volume 50 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*,Springer Berlin Heidelberg, 2012. , pages 326–343.

[3] WILLIAM ENCK, PETER GILBERT, BYUNG-GON CHUN,LANDON P. COX, JAEYEON JUNG, PATRICK MCDANIEL, and ANMOL N. SHETH. TAINTDROID: *an information-flow tracking system for realtime privacy monitoring on smartphones*. In *Proceedings of the 9th USENIX conference on Operating systems design and implementation*, OSDI'10, CA,USA, 2014. , pages 1–6, Berkeley USENIX Association.

[4] J. B. MACQUEEN. *Some methods for classification and analysis of multivariate observations*. In L. M. Le Cam and J. NEYMAN, editors, *Proc. of the fifth Berkeley*

Symposium on Mathematical Statistics and Probability, volume 1, University of California Press, 2013. pages 281–297.

[5] ASAF SHABTAI, URI KANONOV, and YUVAL ELOVICI. *Intrusion detection for mobile devices using the knowledge-based, temporal abstraction method. J.Syst. Softw.*, August 2014. 83:1524–1537.