Process Security – Um *App* para o controle de processos no Android

Gustavo Rotondo[1], Gustavo Amaral[2], Rodrigo Silva[2], Fernando Colosio[2], Érico Amaral[1]

[1]Engenharia de Computação – Universidade Federal do Pampa (UNIPAMPA)
Avenida Maria Anunciação Gomes de Godoy, n°1650 – Bagé – RS – Brasil

[2]Análise e Desenvolvimento de Sistemas - Instituto Federal Sul-Rio-Grandense (IFSUL)
Av. Leonel de Moura Brizola, n° 2501 – Bagé – RS – Brasil

gustavo.rotondo@gmail.com, gustavo.h.amaral@gmail.com, orki2008@gmail.com, ericoamaral@unipampa.edu.br

Abstract. *With the increasing use of the Android operating system, mobile devices have been target of several attacks. These attacks are mainly intended to capture user information. While there are several programs aiming to promote security in the Android platform today, these tools do not realize an analysis at a low level in the system, thus leaving loopholes for applications that run at a lower those that antivirus layer has access can take control the device and capture personal information. The objective of this research is to propose and implement a tool that performs the analysis of processes on the device that has embedded the Android operating system.*

Resumo. Com o crescente uso do sistema operacional Android, os dispositivos móveis têm sido alvo de diversos ataques. Estes ataques têm como objetivo principal a captura de informações do usuário. Embora existam vários programas que visam promover segurança na plataforma *Android*, estas ferramentas não realizam uma análise em baixo nível no sistema, deixando assim brechas para que aplicações que são executadas em uma camada inferior daquelas que o antivírus tem acesso possam ter controle do dispositivo e capturar as informações pessoais do usuário. O objetivo desta pesquisa é propor e implementar uma ferramenta de processos ativos em dispositivos que tenham embarcado o sistema operacional Android.

1. Introdução

Devido ao avanço tecnológico das últimas décadas, o computador tornou–se cada vez mais presente no nosso cotidiano. Nos primórdios da computação tínhamos imensas salas, as quais eram ocupadas por enormes computadores que, se comparados aos atuais, apresentavam um desempenho baixíssimo (SONG,2010). Atualmente contamos com maquinas cada vez menores e com grande poder computacional, dentre eles os chamados *smartphones* que gradativamente tornaram-se mais comuns nos dias de hoje. Com isso os usuários passaram a armazenar suas informações pessoais em seus dispositivos, o que pode se tornar um problema caso o aparelho venha sofrer algum tipo de ataque malicioso.

No universo de dispositivos móveis disponíveis atualmente no mercado, existem diversas plataformas, dentre as quais o sistema *Android* vem se destacando pelo expressivo número de usuários. (NORBEN *et al*. 2013). Dentre os utilizadores de smartphones, 84.7% fazem o uso dessa plataforma, o qual possui aproximadamente 1.3 milhão de aplicações disponíveis para *download* dentre as quais apenas 1% das ferramentas foram consideradas maliciosas.

(HAMMAN, 2014). Por conta do enorme número de usuários e por ser um sistema de código aberto, portanto esta estrutura é uma das mais propícias a ataques de invasores e assim necessita de constantes atualizações e uso de programas para incremento de segurança. (RAMALHO *et. al,* 2013).

Um estudo realizado entre 2010 e 2011 analisou 1260 amostras de possíveis aplicações maliciosas e foi constatado que 86% desses exemplares eram *trojans* e 51% desses programas tinham como objetivo a coleta de informações dos usuários (ZHOU, 2012). Analisando esses dados, podemos concluir que a plataforma *Android* é amplamente utilizada em sua maioria por indivíduos que não possuem conhecimento técnico e devido à isso sofrem constantes ameaças e ataques, o que demonstra a necessidade de alguma ferramenta para o acréscimo de segurança que não faça a varredura somente de aplicações e arquivos que são visíveis ao utilizador, mas sim vasculhar de maneira mais profunda e completa do sistema.

Este extrato de pesquisa segue a seguinte estrutura: a contextualização do tema, na seção de introdução; a segunda seção apresenta um referencial sobre segurança em dispositivos móveis; os trabalhos correlatos estão descritos na seção 3; a seção quatro apresenta a metodologia, com a descrição do planejamento da pesquisa e, descrição da implementação do ProSec, com um detalhamento de todas as características funcionais desta aplicação; os resultados e discussões estão representados na quinta seção e, por fim; na seção seis são apresentadas as conclusões deste estudo.

2. Segurança no Sistema Android

Com a ampla utilização do sistema operacional *Android*, o SO virou alvo de ataques de *cyber* criminosos, que tentam por meios de vírus e outras pragas virtuais obter informações sobre o usuário ou furtar dados pessoais. Um dos recentes problemas foi detectado em agosto de 2013, conhecido como Master Key (BLACK HAT CONFERENCE, 2013). Trata-se de uma falha em que o atacante modifica o arquivo apk[1] e consegue inserir na aplicação, códigos maliciosos que possibilitam a abertura de brechas no sistema, sendo possível o roubo de informações sigilosas. Tal problema afetava em média 900 milhões de aparelhos que rodavam a versão 1.6 ou superior do *Android*.

Mesmo com todas as políticas de envio de aplicações à *Google Play Store*, pesquisadores da Universidade de Indiana desenvolveram uma ferramenta que detecta *softwares* maliciosos para *Android*. Após a varredura, o programa identificou 127.429 aplicações nocivas em diversas lojas de aplicativos. A pesquisa verificou 1.2 milhões de *softwares* em 33 lojas distintas, e um ponto grave observado pelos pesquisadores foi em relação à loja oficial da *Google*, a qual teve 33 mil *apps* reconhecidos como perigosos. (CHEN *et al*. 2015).

3. Trabalhos Correlatos

O desenvolvimento desta pesquisa é baseado no estudo e experimentos sobre o sistema operacional *Android* e suas fragilidades e buscando, desta maneira identificar aplicações que poderiam ser utilizadas por usuários com o intuito de elevar o nível de confiabilidade dos dispositivos móveis. Concomitante a este estudo, alguns trabalhos correlatos foram investigados, à fim de se verificar o estado da arte sobre soluções de segurança para a arquitetura *Android*.

Zhou & Jian (2012), relatam em sua pesquisa intitulada "*Dissecting Android Malware: Characterization and Evolution*", que o rápido crescimento do sistema *Android* aponta para a necessidade de uma evolução nos padrões de segurança da arquitetura. O estudo descreve que diariamente novos *malwares* são criados com o intuito de se propagarem e roubarem

[1] *Android Package*. Instalador de aplicações para dispositivos *Android*

informações pessoais dos dispositivos móveis. Como resultado da pesquisa foi constatado que 86% dos programas maliciosos analisados pelos pesquisadores eram obtidos por meio de aplicações disponíveis na própria loja de aplicativos da *Google*, sendo que destes, 36% se instalavam na raiz do aparelho, onde conseguiam causar danos irreparáveis e 45% faziam uso de ligações e envio de mensagens sem o consentimento do usuário.

4. Metodologia e Implementação

A realização da pesquisa ocorreu de forma exploratória e aplicada, pois teve como objetivo proporcionar um maior entendimento sobre o tema segurança em dispositivos Android e, propor soluções que resolvam os problemas encontrados.

Para a implementação da ferramenta foi proposto um conjunto de etapas formalmente organizadas: A primeira etapa baseou-se no levantamento do referencial teórico; A etapa 2 concentrou-se no estudo/análise do sistema operacional Android, a fim de reconhecer o padrão de execução dos processos neste sistema (processos nativos e de aplicações do usuário); A proposição da solução (ProSec), baseada em um filtro de segurança, ocorreu na terceira etapa; Na quarta etapa realizou-se o levantamento de requisitos e modelagem da aplicação, adotando-se para isto recursos da UML; A construção da aplicação ocorreu na quinta etapa do projeto, utilizando para isso a *IDE Android Studio* e a linguagem de programação Java, além da *Application Programming Interface* (API) 19 (*KitKat* 4.4.2) da plataforma Android, justificada por ser a versão mais difundida deste sistema atualmente. A sexta e a sétima etapas constituíram-se do estudo/elaboração dos testes de funcionalidade e, da coleta de dados e discussões, respectivamente.

A aplicação ProSec possui como finalidade principal o monitoramento das atividades em segundo plano em um dispositivo rodando o sistema operacional Android. O esquema de funcionamento desta ferramenta é apresentado com um infográfico (Figura 01). Basicamente, ao iniciar a execução deste software, o filtro de segurança analisará os processos e os classificará, utilizando para isso um repositório de processos reconhecidos, denominados *Whitelist* (processos não maliciosos) e *Blacklist* (processos referentes a aplicações maliciosas ou com alto risco). O caso 1 representa um sistema sem riscos, o caso 2 representa um dispositivo potencialmente em perigo.

O ProSec quando ativado verifica a existência do repositório *Whitelist*, caso seja a primeira vez que a aplicação é executada em um determinado dispositivo, é iniciada uma conexão com o servidor de repositórios e o mesmo é carregado para o aparelho. Para a transmissão dos dados, referente ao repositório, um padrão de criptografia e utilizado.

O catálogo que porta as funções maliciosas é elaborado no momento em que o mecanismo captura um processo classificado como malicioso, ou seja, que não conste na *Whitelist*. Caso o usuário opte por remover a aplicação capturada pela ferramenta, o processo referente a esta aplicação é registrado na *Blacklist*, desta forma o ProSec não permite que o sistema operacional instale novamente este software. A construção do repositório com os processos maliciosos é realizada em uma única vez, a partir deste ponto, o repositório é apenas atualizado com os novos processos identificados como maliciosos.

Caso o filtro capture um processo que não esteja na *Whitelist* (arquivo com a lista de processos não maliciosos) ou que esteja contido na *Blacklist*, o sistema envia um alerta ao usuário, disponibilizando as seguintes opções: (1) Desinstalar a aplicação, (2) Ignorar ou (3) Exibir detalhes da mesma (permissões e nível de risco).

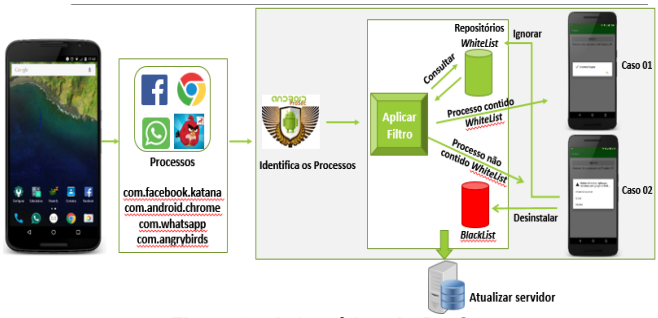


Figura 01. Infográfico do ProSec

Selecionando o modulo de remoção da aplicação (1), será enviado um comando para encerrar o processo capturado e posteriormente a remoção da mesma. A primeira etapa é necessária, pois caso o método apreendido tenha privilégios de administrador, o sistema ficará incapaz de desinstalar a ferramenta.

Quando ativada o modulo Ignorar o procedimento (2), a tarefa incluirá o referente processo diretamente na *Whitelist*, reconhecendo esta como um software seguro. A técnica então será feita para que o filtro de segurança não detecte novamente a dada tarefa.

Ao selecionar a opção Exibição de detalhes (3), o usuário poderá verificar todas as permissões que o software necessita para sua execução. Ainda, neste modulo são disponibilizados um conjunto de funcionalidades: resumo da aplicação capturada (descrição do software), acessos a informações direto da loja *Google Play Store* (informações da loja e opiniões de usuários), análise de riscos do processo (definição do nível de risco da aplicação) e exibição da assinatura do desenvolvedor. Este ultimo recurso permite identificar o responsável pelo desenvolvimento da aplicação, a fim de identificar a veracidade da aplicação, ao comparar esta assinatura com a armazenada na *Google*. Caso estas informações estejam corretas, o processo identificado terá seu software classificado como seguro. A fim de apresentar o funcionamento do ProSec os Casos de Uso e Diagrama de Atividades são apresentados nas Figuras 02 e 03 respectivamente.

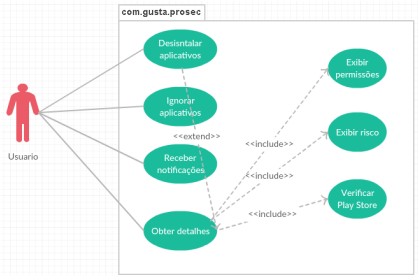


Figura 02. Casos de uso

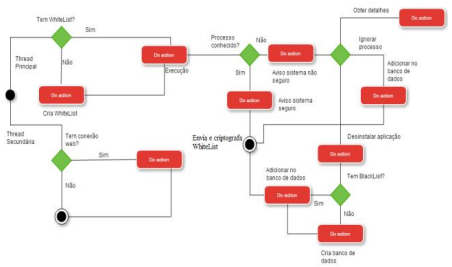


Figura 03. Diagrama de Atividades

Para a analise de riscos das aplicações capturadas pelo ProSec foi proposto um modelo baseado na avaliação das permissões mais comuns entre *malwares* e atribuição de um peso para cada autenticação requisitada. Dentre os acessos mais perigosos, se encontram os privilégios de acesso à *Internet* e obter o *status* da conexão Wifi e do telefone. Ao final da análise das

permissões, os riscos de cada uma destas autorizações eram somados e avaliados. A partir desta técnica o estudo de Jacobsen (2014) mostrou que um *malware* analisado com uma ameaça mensurada em 34, representaria a média aritmética entre os *malwares* pesquisados. Portanto caso um processo capturado apresente um risco maior ou igual a 29, o mesmo é classificado como sendo de alta periculosidade.

A implementação da ferramenta foi feita utilizando a linguagem de programação *Java*, a qual tem suporte nativo por parte da arquitetura *Android* e a interface gráfica foi implementada utilizando os recursos *XML* pela maior compatibilidade com versões mais antigas do sistema e por demandar menos recursos de processamento. Escolheu-se a *IDE Android Studio.* Utilizou-se a versão 2.1 e esta *IDE* foi a escolhida por ter sido feita pela própria *Google*, com isso a ferramenta já vem pronta para que qualquer desenvolvedor comece a programar seus aplicativos. Dentre as *IDE's NetBeans* e *Eclipse*, se fazia necessário a instalação do *Software Development Kit (SDK)*, e isso por sua vez poderia ocasionar mal funcionamento proveniente de passos errados durante a sua instalação.

Com o ambiente de programação pronto para iniciar o desenvolvimento, foram analisadas maneiras de extrair os processos que estavam em execução no dispositivo. Para realizar a listagem dos procedimentos, foram utilizadas estruturas que permitem que comandos sejam executados diretamente no *shell* do sistema *Android*. Como o sistema é oriundo do sistema *Linux*, utilizou-se o comando *top* para listar os processos em execução. O processo é iniciado quando o usuário clica no botão nomeado *btnVerifica*. A função responsável pelo método recebeu o nome de *Executa* e retorna uma *string* a qual é passada para uma área de texto para ser visualizada pelo usuário.

Para que a verificação dos processos fosse feita, utilizou-se o método de comparação de processos exibidos com o arquivo que contém os procedimentos seguros. Ao iniciar a função *Executa*, todo conteúdo da *WhiteList* e da área de texto são copiados para um *vetor* para que as comparações sejam feitas. Quando ocorre uma comparação e a ferramenta identifica que um determinado processo apareceu na área de texto e o mesmo não estava contido na *WhiteList*, uma variável booleana tem seu valor alterado para *true* (verdadeiro) e o método capturado é salvo em um terceiro vetor. Este vetor será utilizado para armazenar os processos classificados como potencialmente maliciosos.

Para sinalizar ao usuário a identificação de um processo suspeito, implementou-se uma classe que permite o envio de notificações. O sistema informa se foi encontrado um procedimento duvidoso caso a variável booleana esteja em um estado *true* (verdadeiro) ou se o sistema estiver seguro caso a variável booleana estiver em estado *false* (falso). É emitido o alerta na forma sonora e visual e através de dispositivos executando o sistema *Android Wear.*

Ao identificar um processo suspeito, uma rotina específica disponibiliza ao usuário um conjunto de ações a serem realizadas para que o utilizador determine qual ação tomar quanto ao método capturado. Como explicado, o *user* poderá ignorar o procedimento, obter detalhes do mesmo ou desinstalar a aplicação geradora do processo tomado. Caso o usuário decida ignorar o processo capturado, o sistema carrega a *WhiteList* e insere o procedimento no arquivo para que a ferramenta não venha alarmar o utilizador. Para a opção de remoção do processo, são utilizados pacotes que utilizam o nome do pacote da *app* capturada para desinstalar a mesma. Ao deletar o programa, o sistema verifica se a *BlackList* existe no dispositivo. Caso o arquivo não exista, é feito a criação do mesmo e posteriormente a inclusão do nome do processo desinstalado. Caso ele exista, somente a parte de inserção é realizada.

A opção de "Obtenção de detalhes" disponibiliza ao usuário o recurso "Exibição permissões" que o mesmo requisita ao sistema, de abrir na *Google Play Store* a página referente ao processo capturado ou realizar uma análise de risco de a aplicação a qual ira atribuir um peso a cada permissão requisitada pelo aplicativo. A escolha da "Exibição Permissões" fornece ao usuário todos os requerimentos permissivos do processo. O sistema exibe estas permissões em uma *ListView* e ao clicar sobre determinada posição, é exibido brevemente na tela uma rápida explicação sobre determinada autorização.

Para o envio do repositório ao servidor, será feito a encriptação desse arquivo. Utilizou-se o método *DES* (*Data Encryption Standard*) o qual se mostrou a melhor opção quando comparado com o método *AES* (*Advanced Encryption Standard*) quando analisado o tempo para realizar a criptografia do arquivo. Ao receber o arquivo de *log*, o servidor *FTP* deverá ser capaz de descriptografar o arquivo recebido, analisar o seu conteúdo, criar um novo arquivo com os processos, e posteriormente enviar para os aparelhos já criptografados.

Criou-se uma classe *SendFTP* e *ReceiveFTP* que implementa a funcionalidade *Web* à ferramenta. O intuito é ter um sistema cliente servidor no qual todos os dispositivos que executem o utilitário *ProSec* e façam o envio de seus arquivos contendo os processos capturados. Ao receber essas informações, um servidor reunirá todos os processos recebidos em dois arquivos denominados *WhiteList* e *BlackList*. Posteriormente o servidor irá reenviar esses arquivos para todos os dispositivos com o intuíto de aumentar a base de dados e assim manter o aplicativo atualizado.

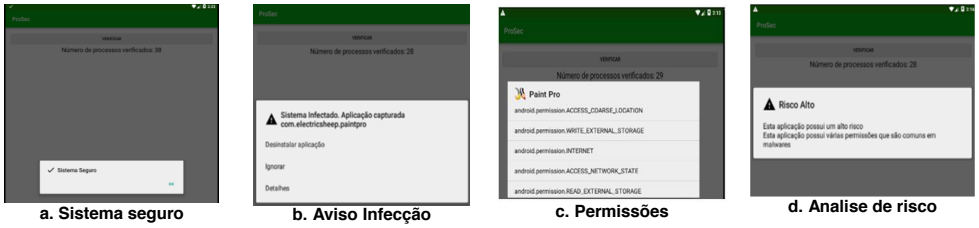
Para evitar que o atacante crie uma aplicação maliciosa e manualmente a insira como processo confiável, será implementado futuramente no servidor *FTP* uma verificação nos arquivos recebidos que consiste em analisar as ocorrências dos procedimentos nos arquivos recebidos. Se um método aparece em uma única *WhiteList* ou em um número reduzido de arquivos, o processo é ignorado e não será incorporado na *WhiteList* final.

5. Resultados e discussões

Esta etapa teve como objetivo a realização de testes do utilitário proposto visando validar as suas funcionalidades. Foram analisados os requisitos funcionais e usabilidade da ferramenta.

Os dispositivos móveis utilizados foram, um aparelho *Motorola Moto X Style* executando a versão 6.0 do sistema *Android* que continha os seguintes programas instalados: *Facebook, WhatsApp, Chrome, Wolfram Alpha, Google Tradutor, AirDroid, Gmail, Google Play Store, Prosec* dentre outros aplicativos nativos do sistema, e um tablet *Gradiente* rodando a versão 5.1.1 da plataforma *Android* que possuia alguns jogos, *Google Play Music, Google Tradutor, VLC, Spotify, UC Browser e Bit Torrent.* Para testes com *malwares*, foi criado uma máquina virtual *Android* executando a versão 4.4.2 do sistema operacional.

O primeiro teste ocorreu em uma máquina virtual rodando a versão 5.0 da plataforma *Android* na qual nenhum aplicativo de terceiros estava introduzido, com isso o resultado para o usuário foi de que o sistema estava seguro (Figura 04a). A segunda rodada dos experimentos consistiu em um ambiente emulado, no qual foi inserida uma aplicação maliciosa. Conforme Wyatt (2011) relatou, novos *malwares* foram encontrados na loja de aplicativos *Google Play Store* naquele ano, entre elas uma *app,* denominada *Paint Master*, a qual infectou cerca de 120 mil dispositivos. O *Paint Master* foi inserido manualmente no sistema e depois ativado o ProSec, que identificou este *malware*, disparando um alerta ao usuário, conforme mostrado na Figura 04b.



a. Sistema seguro **b. Aviso Infecção** **c. Permissões** **d. Analise de risco**

Figura 04. Exibição dos detalhes

A etapa seguinte disponibilizou informações sobre o processo identificado, possibilitando a visualização das permissões requeridas pelo aplicativo capturado e a obtenção de uma análise de risco daquela ferramenta. Os detalhes estão mostrados na Figura 04c. Ao selecionar a opção de detalhes, o usuário pode escolher entre exibir as permissões que o app requisita ao sistema, abrir a aplicação diretamente na Google Play Store, obter a assinatura do programa capturado ou fazer uma análise de risco do software. Não foi possível a abertura do aplicativo na GPS pois a mesma já tinha sido removida pela própria empresa. Ao escolher a alternativa de autenticidades, o utilizador é informado de todos os vistos que o app requer para seu funcionamento e ao selecionar cada uma das autorizações, são exibidas no rodapé do aparelho algumas informações sobre a permissão selecionada.

O caso seguinte consistiu na exibição do nível de ameaça do aplicativo coletado. Esse método faz uma análise das permissões requeridas e informa para o usuário se a aplicação representa algum perigo, podendo estes serem classificados como baixo, médio ou alto. Após a desinstalação do aplicativo capturado realizou-se novamente uma verificação do sistema e constatou-se que o mesmo estava seguro.

A fim de validar a ferramenta ProSec, este projeto seguiu conceitos de testes de software propostos por Rocha (2001), adotando testes Funcionais e de Aceitação.

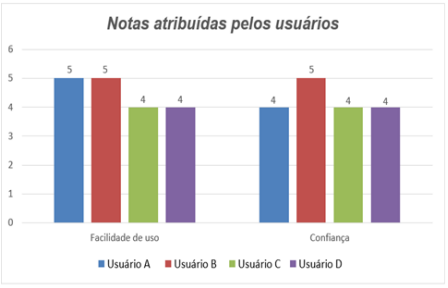


Figura 05 – Notas atribuídas pelos usuários

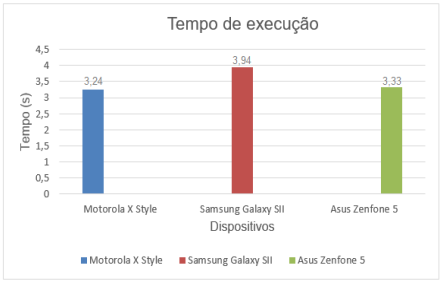


Figura 06 – Tempo de execução do ProSec

Como exemplo dos resultados obtidos com testes de aceitação é apresentado na Figura 05 notas atribuídos pelos usuários, durante os experimentos, sobre a utilização da aplicação. Denota-se que, em uma escala Likert de 5 pontos, a maioria dos usuários descreveu a ferramenta como de fácil uso e que garantia um nível a mais de segurança ao sistema, identificados por médias 4,5 em facilidade e 4,25 para confiança.

Para aferir o desempenho da ferramenta realizou-se um teste de verificação de processos em três dispositivos com configurações de hardware e versões do sistema Android distintos e um teste para verificar o tempo total para concluir o download da *Whitelist*. O teste ocorreu utilizando medições de tempo disponibilizadas pela própria IDE, e foi feita a verificação em cada

dispositivo 3 vezes seguidas. A cada nova rodada de verificações excluiu-se o *app* ProSec. Verificou-se que nas mesmas condições de uso, o dispositivo da Samsung realizou o procedimento em 3,94 segundos. O dispositivo Motorola apresentou um delay de 3,24 segundos para realizar a mesma tarefa e o smartphone Asus apresentou um delay de 3,33 segundos. Levando em consideração que o dispositivo Samsung Galaxy S II, lançado no primeiro trimestre de 2011, não houve grande disparidade de tempo com o dispositivo da Motorola que possui configurações de hardware bem superiores e que foi lançado no último trimestre de 2015. A figura 06 ilustra os diferentes os tempos de execução do aplicativo.

6. Conclusões

Ao identificar a utilização em larga escala dos dispositivos móveis e questões fundamentais de segurança relacionados a este contexto, este projeto avaliou e propôs uma solução para o reconhecimento de possíveis processos maliciosos no sistema, com base em uma estrutura na qual temos armazenado quais processos não representam riscos para o usuário. A retirada de procedimentos ocorreu à partir de uma análise na loja de aplicativos *Google Play Store* e em dispositivos previamente formatados para que fosse possível a coleta de métodos que fossem nativos do sistema.

Após os testes realizados constatou-se que o utilitário ProSec conseguiu de forma eficiente reconhecer processos que não eram conhecidos do sistema e com isso alertar o usuário sobre procedimentos desconhecidos. Quando foi inserido manualmente um *malware* no sistema, além de detectar o método como desconhecido, a aplicação foi capaz também de verificar o risco do *software* a qual foi corretamente classificada como um *app* de alto risco.

O aplicativo que implementa o modelo de análise de processos criado foi concluído com êxito, mantendo bons aspectos de usabilidade e eficiência, comprovados por testes.

Referências

BLACK HAT CONFERENCE “Android Master Key Vulnerability Makes Us Safer”. Disponível em <http://www.esecurityplanet.com/mobile-security/black-hat-android-master-key-vulnerability-makes-us-safer.html>. Acesso em 01/09/2015

CHEN, Kai, et al. "Finding Unknown Malice in 10 Seconds: Mass Vetting for New Threats at the Google-Play Scale." 24th USENIX Security Symposium (USENIX Security 15). USENIX Association.

HAMMAN, “iOS Android e Windows Phone: números dos gigantes comparados”.Acesso em 07/04/2016.

JACOBSEN Wilson, (2014),“ANÁLISE E PROPOSTA DE UM MODELO PARA AVALIAÇÃO DE RISCOS EM APLICATIVOS PARA MÓVEIS”. Bagé – RS

NORBEM “COMPARATIVE EVALUATION OF OPERATING SYSTEMS FOR DEVICES MOBILE: FOCUS ON FUNCTIONALITY”. Disponível em < http://revistas.unifacs.br/index.php/rsc/article/download/2581/1950.>. Acesso em 07/042016

RAMALHO, Abraão Lincon da S.; DOTTO, Roan Siviero; DE OLIVEIRA CARNEIRO, Suliane. ANDROID–O NOVO ALVO DE VÍRUS. In: Workshop de Tecnologia da Região Fronteira Oeste–Anais. 2013.

ROCHA, A. R. C., MALDONADO, J. C., WEBER, K. C. et al., “Qualidade de software – Teoria e prática”, Prentice Hall, São Paulo, 2001.

SONG Wun, “A Evolução dos Computadores: do ENIAC ao Jaguar”, IME/USP (2010). Disponível em < http://www.ime.usp.br/~song/mac412/historia.pdf>. Acesso em 07/04/2016

ZHOU, Yajin; JIANG, Xuxian. (2012), Dissecting android malware: Characterization and evolution. In: Security and Privacy (SP), IEEE Symposium on. IEEE, 2012. p. 95-109.