UNIVERSIDADE FUMEC
FACULDADE DE CIÊNCIAS EMPRESARIAIS - FACE
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ANTÔNIO AUGUSTO DUARTE MOURA DE AQUINO
LEONARDO SILVA PASSARELLA FALCI

A SEGURANÇA DOS SISTEMAS OPERACIONAIS ANDROID

BELO HORIZONTE
2015

ANTÔNIO AUGUSTO DUARTE MOURA DE AQUINO
LEONARDO SILVA PASSARELLA FALCI

A SEGURANÇA DOS SISTEMAS OPERACIONAIS ANDROID

Trabalho de conclusão do curso apresentado à Universidade FUMEC como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Orientador Metodológico: Prof. Msc. Humberto F. Villela

Orientador Temático: Prof. Esp. Davis Anderson Figueiredo

BELO HORIZONTE
2015

ANTÔNIO AUGUSTO DUARTE MOURA DE AQUINO
LEONARDO SILVA PASSARELLA FALCI

Trabalho de Conclusão de Curso defendido e aprovado/reprovado, em: _____/_____/2015, pela bancada examinadora constituída por:

_____
Prof. Msc. Humberto Fernandes Villela – Orientador Metodológico

_____
Prof. Esp. Davis Anderson Figueiredo – Orientador Temático

_____
Prof.

BELO HORIZONTE
2015

RESUMO

Atualmente, o Android é o líder do mercado de sistemas operacionais mobile e é um código aberto utilizado por vários fabricantes, por isto, segundo alguns autores, apresenta falhas que variam de uma marca de produtos para a outra, sendo assim este projeto se limita a analisar apenas as falhas que afetam todos os aparelhos, independente desta variedade. Principais vulnerabilidades encontradas estão relacionadas a certas falhas do Android ao lidar com aplicativos, bibliotecas, ou até mesmo enraizadas no próprio sistema operacional. Estas falhas normalmente são exploradas por desenvolvedores maliciosos visando o roubo de conteúdo no aparelho atacado, manipulação de dados e acesso a informações confidenciais, sendo que esta ação dos criminosos em algumas ocasiões é até facilitada pelo usuário quando este utiliza aplicativos de origem desconhecida, fora da loja oficial do Google. Esta pesquisa surge a partir do questionamento: quais as principais vulnerabilidades que foram reportadas no sistema Android e os seus mecanismos para garantir a segurança? Seu objetivo principal foi realizar estudo sobre as principais vulnerabilidades do sistema operacional Android entre a API 18 e a API 22 (exceto API 20, por ser desenvolvida exclusivamente para dispositivos vestíveis). O estudo focou em analisar o funcionamento do sistema Android e avaliar a autenticidade de aplicativos da Google Play Store (loja virtual onde o usuário pode comprar músicas, filmes, livros e aplicativos), ressaltando suas vulnerabilidades apresentadas por estes sistemas e como as fabricantes que os utilizam em seus dispositivos e a Google lidam com estes problemas. Considera-se que o objetivo da pesquisa foi atingido e que esta gera novos questionamentos que poderão ser foco de futuras pesquisas como analisar a usabilidade em diferentes dispositivos Android e o que isto pode influenciar na segurança do sistema.

Palavras-chaves: Android, Vulnerabilidade, Segurança, Mobile.

ABSTRACT

Currently, Android is the leader in the mobile OS market and is an open source used by several manufacturers, therefore, according to some authors, it has flaws that vary from one product brand to another, therefore this project is limited to analyze only failures affecting all devices, regardless of this variety. Main vulnerabilities found are related to certain failures in dealing with Android applications, libraries, or even embedded in the operating system itself. These flaws are often exploited by malicious developers targeting the content theft wholesale unit, data manipulation and access to confidential information, and this action of criminals sometimes even facilitated by the user when the use of unknown origin applications, out of Google's official store. This research arises from the question: what are the main vulnerabilities that have been reported in the Android system and its mechanisms to ensure safety? Its main objective was to conduct study on key vulnerabilities of the Android OS between API 18 and API 22 (except API 20, being developed exclusively for wearable devices). The study focused on analyzing the operation of the Android system and evaluate the authenticity of the Google Play Store applications (virtual store where you can buy music, movies, books and apps), highlighting their vulnerabilities presented by these systems and how manufacturers that use in their devices and the Google deal with these problems. It is considered that the objective was reached and that this generates new questions that may be the focus of future research to analyze the usability in different Android devices and this can influence the system safety.

Keywords: Android, Vulnerability, Security, Mobile.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AOSP	*Android Open Source Project* (tradução livre: Projeto Código Aberto Android).
API	*Application Programming Interface* (Interface de Programação de Aplicativos).
APK	*Android Application Package Kit* (Pacote instalador específico para Android).
App	Aplicativo.
AVRCP	*Audio/Video Remote Control Profile* (Perfil de controle remoto de áudio/vídeo).
CVE	*Common Vulnerabilities and Exposures* (Vulnerabilidades e Exposições Comuns).
DVM	*Dalvik Virtual Machine* (Máquina Virtual Dalvik).
EXE	Arquivo executável do sistema operacional Microsoft Windows.
GPS	*Global Positioning System* (Sistema de Posicionamento Global).
iOS	*iPhone Operating System* (Sistema Operacional do iPhone).
IP	*Internet Protocol* (Protocolo de Internet).
JVM	*Java Virtual Machine* (Máquina Virtual Java).
MB	MegaBytes.
MITM	*Man-in-the-middle attack.*
MMS	*Multimedia Messaging Service* (Serviço de Mensagens Multimídia).
NDA	*Non-Disclosure Agreement* (Acordo de Não Divulgação).
NFC	*Near Field Communication* (Comunicação por Campo de Proximidade).
OTA	*Over The Air* (“Pelo Ar”; Método de atualização).
RAM	*Random Access Memory* (Memória de acesso aleatório).
SO	Sistema Operacional.
TCP	*Transmission Control Protocol* (Protocolo de Controle de Transmissão).
TI	Tecnologia da Informação.

SUMÁRIO

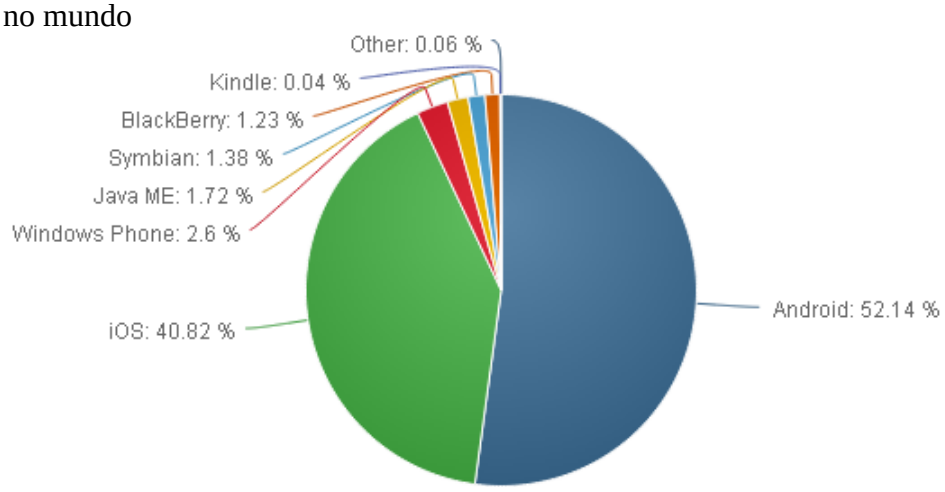
1 INTRODUÇÃO

Com o avanço tecnológico, a humanidade é capaz de criar constantemente novos métodos e habilidades que possibilitam a melhoria das tarefas do cotidiano. Por exemplo, antigamente, para enviar uma carta era necessário postá-la pelo correio e a chegada ao seu destino poderia levar semanas, e hoje, ela pode ser enviada em poucos segundos por meio de um serviço de correio eletrônico (e-mail), contando com a ajuda da Internet.

Atualmente, os tablets e smartphones estão cada vez mais presentes na vida das pessoas e a quantidade de usuários de dispositivos móveis também está se expandindo. A demanda por esses equipamentos está aumentando consideravelmente no mundo inteiro, graças aos seguintes fatores: barateamento da tecnologia, crescente desenvolvimento *mobile* de diversos aplicativos específicos, aperfeiçoamento do poder de processamento de dados dos dispositivos móveis, entre outros. Porém, essa demanda acaba forçando a necessidade da adoção de sistemas operacionais que atendam a variedade de dispositivos diferentes e ofereçam o maior número de recursos possíveis.

Dentre os principais sistemas operacionais *mobile* existentes no mercado, podem ser citados o Android da Google, o iOS da Apple e o Windows Phone da Microsoft. De acordo com dados da Net Market Share (2015), uma respeitada empresa que realiza levantamentos estatísticos com base na análise dos acessos a sites, é possível perceber por meio da Figura 1 que o Android é o sistema operacional *mobile* mais usado do mundo, com 52,14% do mercado. Em seguida, vem o iOS e o Windows Phone, com 40,82% e 2,60%, respectivamente. Os dados mais recentes são de agosto de 2015.

FIGURA 1 - Gráfico de desempenho dos sistemas *mobile* mais utilizados no mundo



Fonte: NET MARKET SHARE, 2015

Com o crescimento constante da venda de dispositivos móveis, não demorou muito tempo para que eles ultrapassassem a quantidade da população mundial. Conforme os dados estatísticos em tempo real da empresa GSMA (2015), atualmente existem cerca de 7,6 bilhões de dispositivos em uso. Além disso, um estudo levantado pela Cisco (2014) estima que em 2019, aproximadamente 11,5 bilhões de dispositivos estarão conectados à Internet simultaneamente. É algo impressionante que a humanidade nunca esteve tão conectada e comunicativa.

1.1 Contextualização do problema

A fragilidade dos sistemas informatizados não é novidade. Existe uma série de fatores que deixam um equipamento exposto a incontáveis riscos. O fato da popularização rápida do sistema Android tem despertado a atenção e o interesse dos cibercriminosos a desenvolverem aplicativos maliciosos para essa plataforma que podem interferir no funcionamento correto do dispositivo móvel e roubar qualquer tipo de dados e informações armazenadas no equipamento sem que o usuário perceba. Outro problema do sistema Android é que, por se tratar de um sistema de código e plataforma abertos, o usuário tem a opção de liberar as permissões de um super-administrador, o que é extremamente arriscado, já que se o dispositivo com acesso ao *root* for infectado, o aplicativo malicioso poderá assumir todo o controle do sistema e fará tudo o que o cibercriminoso deseja.

Por causa desse e vários motivos que deixam o Android vulnerável a falhas, muitas questões relacionadas à Segurança da Informação devem ser revisadas e esclarecidas para que os usuários consigam utilizar seus equipamentos da forma mais segura possível, assim evitando supostas ameaças, invasões e problemas de vulnerabilidades no sistema. Afinal, quais as vulnerabilidades mais comuns que foram reportadas no sistema Android e os seus mecanismos para garantir a segurança?

1.2 Objetivo Geral

Este trabalho consiste num estudo sobre as principais vulnerabilidades do sistema operacional Android entre a API 18 e a API 22 (exceto API 20).

1.3 Objetivos Específicos

- Avaliar a autenticidade dos aplicativos da Google Play Store;

- Identificar e compreender o funcionamento das atualizações e correções para os sistemas Android;
- Identificar, analisar e comparar as versões 4.3, 4.4, 5.0 e 5.1 do Android;
- Analisar as principais vulnerabilidades reportadas no uso do sistema operacional Android.

1.4 Justificativa

Pretende-se descobrir o que o sistema Android oferece em termos de segurança aos usuários e fazer uma busca das principais vulnerabilidades e problemas encontrados, a fim de entender como a segurança no sistema Android deve ser tratada.

Este trabalho se justifica em três ênfases. A primeira, sob a ótica técnica ou científica, onde pretende-se avançar na discussão do tema central segurança e sua relação direta com o sistema operacional Android.

Empresarialmente, o tema e trabalho se justificam diante da relevância do sistema operacional Android e sua parcela de mercado e uso atualmente. Faz-se necessário discutir suas vulnerabilidades e características técnicas relacionadas à segurança frente a diversas aplicações em uso pelas organizações e usuários finais.

No cenário atual, não existe um sistema considerado 100% seguro à prova de falhas e de vulnerabilidades, porém é necessário evitar os riscos da insegurança. Entende-se que o Android é o alvo preferido dos cibercriminosos que visam atacar e invadir dispositivos móveis, através do desenvolvimento de aplicativos maliciosos e que também se aproveitam das falhas de segurança de uma versão específica do Android.

A terceira argumentação da justificativa tem cunho na equipe que o desenvolveu o estudo. O trabalho refere-se ao Trabalho de Conclusão de Curso de Bacharelado em Ciência da Computação onde buscou-se avançar nos conhecimentos técnicos obtidos ao longo do referido curso e ainda alinhamento ao mercado de empresarial. Entende-se que o sistema operacional Android deve ser investido e utilizado por diferentes tipos de aplicações e organizações sendo foco o mercado de trabalho dos estudantes envolvidos.

1.5 Hipótese

Ao realizar esta pesquisa, tem-se a hipótese de que as vulnerabilidades não impactam no uso do sistema Android.

Espera-se confirmar que as vulnerabilidades são entendidas como dificuldades de proteger informações do sistema, embora sua liberdade de uso e ação dos usuários nos dispositivos móveis que utilizam a referida tecnologia são atrativos que podem ser justificativas para investimento em Segurança da Informação.

2 REFERENCIAL TEÓRICO

2.1 A importância da segurança nos dispositivos móveis

Da mesma forma que os desenvolvedores e fabricantes criam novos dispositivos móveis, com inovações cada vez mais atrativas e tecnológicas, os cibercriminosos também acompanham as tendências e as evoluções. Logo, novos aplicativos maliciosos são produzidos todos os dias para a tentativa de quebra de segurança.

Apesar da existência de leis para esse tipo de risco, há outros desafios relacionados à segurança do setor *mobile* que não dependem apenas de autoridades, mas, principalmente, de usuários e empresas de tecnologia.

2.2 Os principais tipos de cibercriminosos

Infelizmente, a Internet possui um ambiente com criminosos cibernéticos que procuram roubar informações pessoais, sabotar computadores e converter a experiência de navegação dos usuários num pesadelo. Mesmo com a variedade de softwares de segurança e as precauções apropriadas que ajudam a prevenir essas tentativas, ainda não é o suficiente, principalmente porque nenhum sistema é completamente seguro. Porém, o que deve ser feito é diminuir os riscos e suas possíveis ameaças.

Engana-se quem pensa que só existe a palavra “hacker” no universo do crime digital. Muitos hackers de computadores, com tendências maliciosas ou não, são caracterizados por um conjunto de termos e conceitos específicos. Através do glossário criado por Raymond (1996), que traz as gírias, os jargões, o folclore, o estilo de falar e escrever, o modo de vestir, o tipo de educação e as características de personalidade dos hackers, os principais tipos de cibercriminosos estão divididos nas seguintes categorias: Hackers, Crackers, Phreakers, *White Hat*, *Black Hat* e *Grey Hat*.

2.2.1 Hacker

A definição do termo “Hacker” varia de socialmente muito positiva ou indivíduos talentosos (*White Hat*) a criminosa (*Black Hat*). De acordo com o glossário de Raymond (1996), o termo pode ser definido como:

- Uma pessoa que gosta de explorar os detalhes de sistemas programáveis e esticar suas capacidades, em oposição à maioria dos usuários, que preferem aprender apenas o mínimo necessário.

- Alguém que programa entusiasticamente (até de forma obsessiva) ou que gosta de programar em vez de apenas teorizar sobre programação.

- Uma pessoa capaz de apreciar o valor do *hacking*.

- Uma pessoa que programa bem e rápido.

- Um especialista em um programa específico, ou que trabalha com ou sobre esse programa.

- Um especialista ou entusiasta de qualquer tipo. Ele pode ser um hacker em astronomia, por exemplo, aquele que gosta do desafio intelectual de superar ou contornar limitações.

- Um intrometido malicioso que tenta descobrir informações sensíveis testando falhas de segurança. Todo desenvolvedor que tem a malícia de prejudicar usuários e empresas é considerado como um cracker.

2.2.2 Cracker

Segundo Raymond (1996), há uma diferença bastante significativa entre um hacker e um cracker. Enquanto o hacker sem tendência maliciosa visa tornar a informática acessível a todos e apenas apontar possíveis falhas de um sistema, o cracker é o termo usado para designar um desenvolvedor que pratica a quebra (*cracking*) de um sistema de segurança de forma ilegal ou sem ética, procurando lucrar o máximo possível com a ação.

Além disso, Santos (2010) afirma que, normalmente, o cracker é especializado em piratear softwares comerciais e usa seus conhecimentos para invadir sites e computadores com objetivos ilícitos, como vandalismo ou roubo. Os crackers são excelentes programadores e podem criar programas que infectem ou destruam completamente sistemas alheios sem deixar vestígios, e caso haja algum imprevisto, ele tem noção suficiente para se “esconder”.

Apesar de alguns hackers irem de encontro à lei, eles são movidos pela intenção de promover o conhecimento e o auxílio a terceiros, mas nunca de autopromoção ou destruição do trabalho alheio.

2.2.3 Phreaker

Raymond (1996) também destacou o phreaker, combinação das palavras inglesas *phone* (telefone) e *freak* (maluco), é um hacker especializado em telefonia fixa e/ou móvel, ou seja, é o uso indevido de linhas telefônicas.

Ulbrich (2004, p. 30) também afirma que, no passado, os phreakers empregavam gravadores de fita e outros dispositivos para produzir sinais de controle e enganar o sistema de telefonia, como realizar qualquer chamada sem pagar por ela. Conforme as companhias telefônicas foram reforçando a segurança, as técnicas tornaram-se mais complexas. Atualmente, o *phreaking* é uma atividade elaborada que poucos hackers dominam.

2.2.4 White Hat

White Hat (chapéu branco) é um hacker que estuda sistemas de computação à procura de falhas na sua segurança, mas respeitando os princípios da ética. Ao encontrar uma falha, os hackers *White Hat* normalmente a comunica em primeiro lugar aos responsáveis pelo sistema para que tomem as medidas cabíveis. Muitos hackers *White Hat* desenvolvem suas pesquisas como professores de universidade ou empregados de empresas de informática (RAYMOND, 1996).

2.2.5 Black Hat

Black Hat (chapéu preto) é um hacker que não respeita a ética e usa seu conhecimento de computadores e outras tecnologias para fins criminosos ou maliciosos. Também pode ser chamado de *dark-side hacker* (hacker do lado negro), em referência à série de filmes Star Wars. Geralmente, os hackers *Black Hat* são considerados como crackers (RAYMOND, 1996).

Segundo Martins (2015), os hackers *Black Hat* podem infligir muitos danos em ambos os usuários de computadores e grandes organizações ao roubar informações sigilosas, comprometendo a segurança de grandes sistemas.

Às vezes, um hacker *Black Hat* emprega métodos em que não se utiliza o computador para obtenção de dados, como fazer um telefonema e assumir uma identidade, a fim de obter a senha de um usuário específico.

2.2.6 Grey Hat

Grey Hat (chapéu cinza) é um hacker intermediário entre *White Hat* e *Black Hat*, ou seja, um hacker que invade sistemas por “diversão”, mas que evita causar sérios danos aos sistemas computacionais e que não copia dados confidenciais (RAYMOND, 1996).

Segundo Russo (2013), um hacker *Grey Hat* não trabalha para seu próprio ganho pessoal ou para causar uma carnificina mundial, mas podem tecnicamente cometer certos crimes virtuais e realizar alguns feitos extremamente antiéticos. Por exemplo, o hacker *White Hat* primeiramente pede permissões à corporação ou empresa antes de testar a segurança de sites, softwares ou sistemas. Caso descubra alguma falha em sua exploração o mesmo alerta sigilosamente todos os envolvidos após comprometê-los. Já o hacker *Grey Hat* não utiliza o seu acesso indevido para fins maléficos, mas caso ele invada um sistema de segurança, o mesmo já está comprometido, fato que torna essa ação totalmente ilegal.

2.3 Os principais tipos de aplicativos maliciosos

Um aplicativo malicioso, também conhecido por malware (*malicious software*), é um software destinado a se infiltrar em um computador alheio de forma ilícita, com o intuito de causar algum dano ou roubo de informações, sejam confidenciais ou não (TECHTERMS, 2015).

Entre os principais tipos de malwares se incluem: Adware, Cavalo de Troia, Spyware, Vírus e Worm.

2.3.1 Adware

Adware, união das palavras *advertising* (publicidade) e software, é um programa que mostra propagandas e anúncios na tela sem autorização do usuário, diminuindo o desempenho do computador e deixando a conexão com a Internet mais lenta. Basta exibir uma propaganda ou o usuário acessar tal publicidade que o desenvolvedor do programa estará ganhando dinheiro com isso, mesmo que o usuário não esteja gastando com o produto (BEAL, 2015).

Normalmente, assumem o formato de pop-up, isto é, janelas incômodas que abrem a todo instante enquanto se navega em um determinado site.

2.3.2 Cavalo de Troia

Cavalo de Troia (*Trojan Horse*) é um tipo de programa malicioso que pode se infiltrar num sistema disfarçado como um programa comum e legítimo. Ele serve para possibilitar a abertura de uma porta de forma que usuários com más intenções possam invadir o dispositivo (UOL, 2013).

O *Trojan* faz analogia à história, onde os gregos presentearam os troianos com um enorme cavalo de madeira, tendo escondido seu exército no interior do cavalo para abrir os portões e fazer a invasão. E segue essa mesma filosofia. Trata-se de um arquivo aparentemente inofensivo, porém com um código malicioso inserido em seu contexto (GRIFFIN, 2000).

Ishimi (2005) também afirma que um *Trojan* se passa por um programa que simula alguma funcionalidade útil, quando de fato ele esconde um programa que pode causar malefícios aos computadores e seus usuários, como abrir portas de rede e possibilitar invasões ou roubar senhas de usuários. Por exemplo, pode-se ter um jogo eletrônico que funciona corretamente, mas que de forma oculta executa algum código malicioso que rouba senhas ou altera as configurações do sistema.

2.3.3 Spyware

Spyware é um programa “espião”, isto é, um software que pode se instalar ou executar no computador sem que o usuário tenha conhecimento ou sem o seu consentimento ou controle. O spyware pode não apresentar sintomas depois de infectar o computador, mas muitos tipos deles podem afetar o funcionamento do computador. Por exemplo, um spyware pode monitorar o comportamento online do usuário ou coletar informações sobre ele, inclusive informações confidenciais ou de identificação pessoal, alterar configurações do computador ou fazer com que ele fique lento (MICROSOFT, 2015).

Todavia, isto não significa que eles sejam em sua totalidade programas maus. Existem muitos spywares de má índole, criados para coletar informações pessoais e, com elas, praticar atividades ilegais. Entretanto, nem todos são assim. Por exemplo, existem empresas de anúncio que utilizam spywares para, de forma legal, coletar informações de seus assinantes, com vistas a selecionar o tipo de anúncio que irão lhes apresentar (XAVIER, 2008).

O fato é que não existe um modo de saber qual spyware é inofensivo ou perigoso. O critério a ser adotado para se proteger é sempre desconfiar. Um spyware não prejudicial só será instalado mediante a autorização do usuário. Porém, um spyware maligno irá se instalar sem que o usuário perceba (UOL, 2013).

2.3.4 Vírus de computador

Segundo a Info Wester (2013), vírus de computador é um programa com fins maliciosos capaz de causar transtornos com os mais diversos tipos de ações, como apagar ou alterar arquivos dos usuários, prejudicar o funcionamento do sistema operacional danificando ou alterando suas funcionalidades, causar excesso de tráfego em redes, entre outros.

Os vírus, tal como qualquer outro tipo de malware, podem ser criados de várias formas. Os primeiros foram desenvolvidos em linguagens de programação como C e Assembly. Hoje, é possível encontrar inclusive ferramentas de desenvolvimento que auxiliam na sua criação.

Os vírus recebem esse nome porque possuem características de propagação que lembram os vírus reais, isto é, biológicos. Quando um vírus contamina um computador, além de executar a ação para o qual foi programado, tenta também se espalhar para outras máquinas, tal como fazem os vírus biológicos nos organismos que invadem.

Antigamente, os vírus tinham um raio de ação muito limitado: se propagavam, por exemplo, toda vez que um disquete contaminado era lido no computador. Com o surgimento da internet, no entanto, essa situação mudou drasticamente, para pior. Isso acontece porque, com a Internet, os vírus podem se espalhar de maneira muito mais rápida e contaminar um número muito mais expressivo de computadores. Para isso, eles podem explorar vários meios, por exemplo, falhas de segurança de sistemas operacionais e programas, envio de e-mails falsos para usuários e downloads de arquivos contaminados.

2.3.5 Worm

Worm (verme) possui as mesmas particularidades de um vírus de computador, porém sua diferença tem como objetivo a disseminação através da rede de computadores e não entre arquivos de um sistema operacional, como ocorre naturalmente nos vírus (BORGES, 2006).

Os worms não precisam obrigatoriamente de um portador para se propagar. Eles podem se autorreplicar, espalhando-se de um computador para outro e ainda pode conter vírus para

infectar os sistemas. Também exploram as vulnerabilidades dos serviços e utilizam quaisquer mecanismos para se propagarem, como TCP/IP, e-mail, serviços de Internet, compartilhamento de arquivos, mídias removíveis, entre outros (GASPAR, 2007).

A partir disso, o worm pode tornar o dispositivo infectado vulnerável a outros ataques e provocar danos apenas com o tráfego gerado pela sua reprodução (UOL, 2013).

2.4 A origem do sistema operacional Android

Tudo começou na empresa Android Inc., fundada por Andy Rubinera, Nick Sears e Chris White em outubro de 2003, na cidade de Palo Alto, Califórnia, Estados Unidos. Inicialmente, a empresa desenvolvia sistemas operacionais portáteis e todos os seus projetos eram secretos, sem a participação de outras companhias. Dentre os seus principais objetivos, estava o desenvolvimento de um sistema operacional avançado para câmeras digitais. Algum tempo depois, percebeu-se que o mercado para tais equipamentos não era grande suficiente, então a equipe desviou os seus esforços para produzir um sistema operacional para smartphones, rivalizando assim com outros sistemas da categoria na época, como o Symbian, desenvolvido pela Nokia, e o Windows Mobile, da Microsoft. Entretanto, a falta de investimentos impossibilitava o bom andamento do projeto (MEYER, 2015).

Em agosto de 2005, a Google anunciou a compra da Android Inc. por US$ 50 milhões. Esse foi um dos primeiros passos da empresa em direção ao mercado de softwares para dispositivos móveis, que culminaram, em novembro de 2007, no lançamento do projeto intitulado “Android”, cujo objetivo era criar uma plataforma de desenvolvimento para dispositivos móveis que contém um sistema operacional sob o padrão aberto[1] baseado no *Kernel* do Linux versão 2.6. Com as novas patentes adquiridas, o projeto Android passou a fazer parte do “Open Handset Alliance[2]”, um consórcio de empresas de tecnologia composta por empresas como a Samsung, Sony, HTC, Motorola, LG, operadores de telefonia e fabricantes de dispositivos, pelo qual é liderado pela Google (MEYER, 2015).

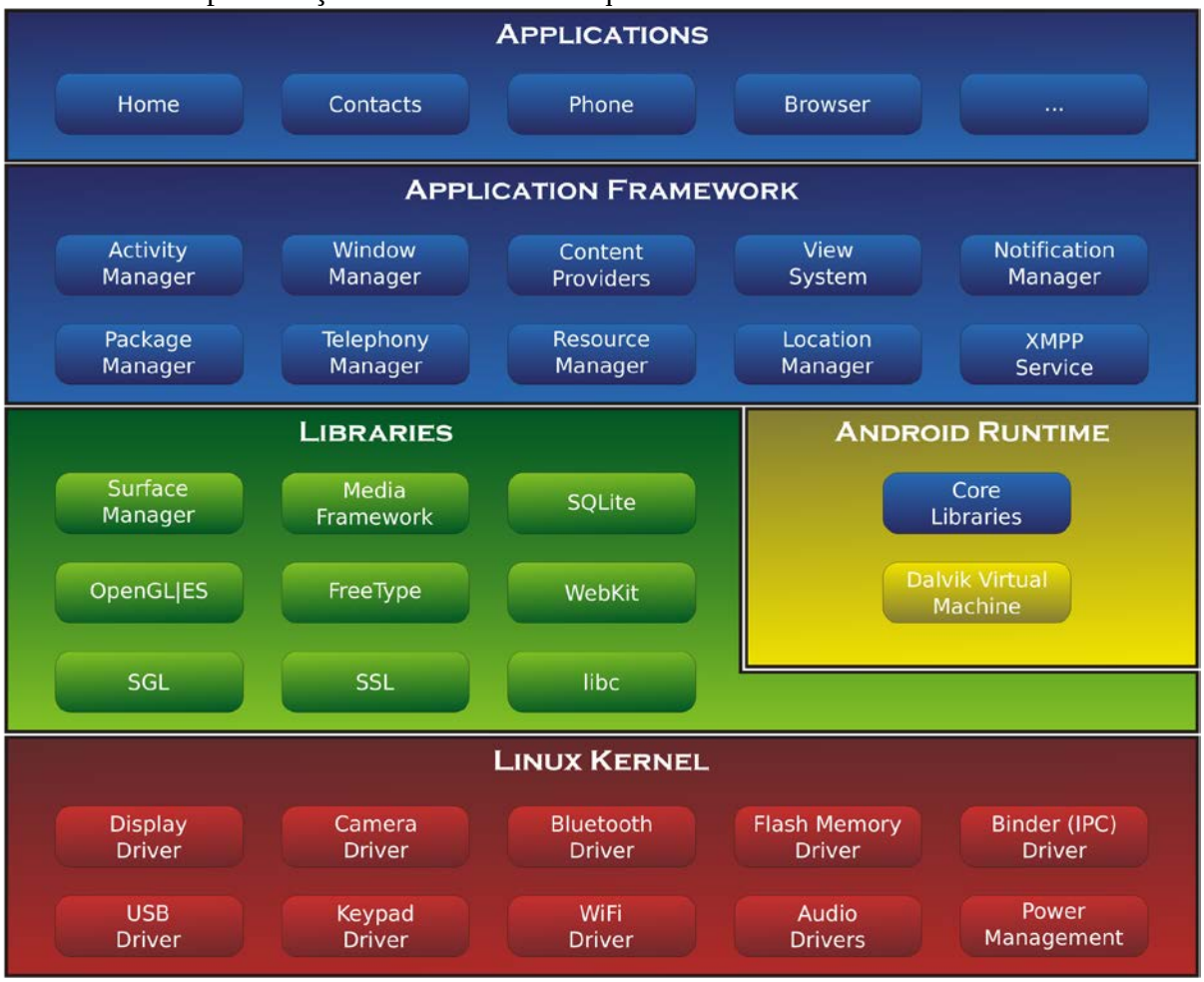
O primeiro equipamento com o sistema operacional Android instalado foi o HTC Dream, lançado em 22 de outubro de 2008 nos Estados Unidos (LECHETA, 2015, p. 32).

[1] Padrão que está disponível ao público e tem vários direitos de uso associado, e também pode ter várias propriedades de como foi projetado.
[2] Lista completa de empresas pode ser conferida através do site: http://www.openhandsetalliance.com/oha_members.html

2.5 Arquitetura do Android

Conforme Lecheta (2015), a arquitetura do sistema Android é dividida em 5 camadas. São elas, começando pela camada mais baixa até a mais alta: *Linux Kernel*, *Libraries*, *Android Runtime*, *Application Framework* e *Applications*. Cada camada é responsável por gerenciar os seus próprios recursos, como é explícito na figura 2.

FIGURA 2 - Representação das camadas da arquitetura do Android



Fonte: LECHETA, 2015

2.5.1 Camada “Linux Kernel”

A camada *Linux Kernel* (núcleo do Linux) é a base da plataforma Android. Para desenvolvê-la foi utilizada a versão 2.6 do sistema operacional Linux. Esta camada atua também como responsável pela abstração entre o hardware e os aplicativos e é responsável pelos serviços principais do sistema operacional Android, como o gerenciamento de memória e de processos, configurações de segurança e vários drivers de hardware. Várias funções do *Kernel* são utilizadas diretamente pelo Android, mas muitas modificações foram feitas para otimizar

memória e tempo de processamento das aplicações. Essas modificações incluem novos dispositivos de drivers, adições no sistema de gerenciamento de energia e um sistema que possibilita terminar processos de maneira criteriosa quando há pouca memória disponível. O Linux 2.6 foi escolhido por já conter uma grande quantidade de drivers de dispositivos sólidos e por ter um bom gerenciamento de memória e processos (GOMES, 2012).

2.5.2 Camada “Libraries”

A camada *Libraries* (Bibliotecas) é um conjunto de instruções que dizem ao dispositivo como lidar com diferentes tipos de dados, incluindo um conjunto de biblioteca C/C++ usadas por diversos componentes do sistema e são expostas a desenvolvedores através da estrutura de aplicativo Android. Exemplos desses componentes são: uma implementação da biblioteca padrão do C (libc), porém com licença BSD e otimizada para dispositivos embarcados; bibliotecas para suporte a formatos de áudio, vídeo e imagens; um gerenciador que intermedia o acesso ao *display* e compõe as camadas de imagem 2D (SGL) e 3D (OpenGL ES); o *engine* para navegadores WebKit; um renderizador de fontes bitmap e vetoriais; o banco de dados relacional SQLite (BORDIN, 2012).

2.5.3 Camada “Android Runtime”

A camada *Android Runtime* (tempo de execução do Android) está divida em duas partes. O *Core Libraries* inclui um conjunto de bibliotecas do núcleo Java que fornece a maioria das funcionalidades disponíveis, como estruturas de dados, acesso a arquivos, acesso a redes e gráficos (PEREIRA, 2012). Nessa camada também se encontra a Máquina Virtual Dalvik (DVM), onde são executados os processos da aplicação. A DVM executa os arquivos em um formato otimizado para o baixo consumo de memória, que são os *Dalvik Executables* (.DEX), resultantes da compilação dos aplicativos Android (MACK, 2010).

O Android usa a DVM para executar cada aplicação com seu próprio processo. Isso é importante por algumas razões: nenhuma aplicação é dependente de outra e se uma aplicação parar, ela não afeta quaisquer outras aplicações executadas no dispositivo e isso simplifica o gerenciamento de memória, porque a DVM está baseada em registradores e desenvolvida de forma otimizada para requerer pouca memória e permitir que múltiplas instâncias executem ao mesmo tempo (CÁRDENAS, 2011).

Nas versões anteriores ao Android 5.0 *Lollipop*, cada uma dessas aplicações no sistema funcionava em uma instância diferente da DVM. Porém, a partir dessa mesma versão, a DVM foi substituída pela *Android Run Time* (ART), que inclusive foi introduzida opcionalmente no Android 4.0 *KitKat*. Enquanto a DVM utiliza *just-in-time compilation*, compilando trechos do código para execução nativa em tempo de execução, a ART introduz o *ahead-of-time compilation*, realizando compilações em tempo de instalação. Embora a instalação possa levar mais tempo dessa forma, essa mudança permite que os aplicativos tenham maior performance em sua execução. Esse isolamento de aplicativos, onde cada um é executado em sua própria instância da máquina virtual, permite que uma falha em um processo de uma aplicação não tenha impacto algum em outra aplicação (PEREIRA JÚNIOR, 2014).

Para desenvolver aplicações para o Android, os programadores utilizam a linguagem de programação Java. Apesar de existir algumas semelhanças, a DVM e a ART não tem nenhuma relação com a Máquina Virtual Java (JVM), porque elas executam o seu próprio tipo de *bytecode*. Na verdade, o que se tem é uma máquina virtual otimizada para execução em dispositivos móveis. Além disso, os desenvolvedores e os usuários não são afetados se o sistema Android está utilizando a DVM ou ART, mas, segundo a Google (2013), o ART apresenta um desempenho muito melhor (LECHETA, 2015).

2.5.4 Camada “Application Framework”

A camada *Application Framework* (Framework de Aplicação) envolve programas que gerenciam as funções básicas do dispositivo móvel, como alocação de recursos, aplicações de telefone, mudança entre processos ou programas. Possibilita aos desenvolvedores o total acesso ao framework para que tirem vantagens das capacidades do hardware do dispositivo, execução de serviços em background, definição de alarmes e muitos outros (MACK, 2010).

Todos os conjuntos de componentes disponíveis no sistema Android possuem pleno acesso às APIs utilizadas pelo núcleo da plataforma. Esse framework foi projetado para simplificar a reutilização de componentes e assim criar ferramentas mais complexas a partir de ferramentas básicas (GOMES, 2010).

2.5.5 Camada “Applications”

A camada *Applications* (Aplicativos) é a camada de interação entre o usuário e o dispositivo móvel. Nela, estão envolvidos os aplicativos como cliente de e-mail, programa de mensagens

de texto e multimídia (SMS e MMS), calendário, mapas, navegadores de Internet, contatos, calculadora, jogos, entre outros.

2.6 Histórico de versões do Android

Até o momento, a Google já produziu 56 versões oficiais do Android. A versão mais recente é a 6.0 *Marshmallow*, lançada no dia 5 de outubro de 2015.

Algo curioso sobre o Android é que algumas de suas versões são apelidadas carinhosamente com o nome de um doce em ordem alfabética. Isso gera sempre uma grande expectativa e especulação no mercado, porque todos ficam tentando adivinhar qual será o novo sabor do Android (LECHETA, 2015).

A tabela 1 apresenta a evolução histórica do Android desde o seu lançamento, em 2008.

TABELA 1 - Histórico de versões oficiais do Android lançadas pela Google

Versão	**API**	**Codinome**	**Lançamento**
1.0	1	–	23 set. 2008
1.1	2	–	09 fev. 2009
1.5	3	Cupcake	27 abr. 2009
1.6	4	Donut	15 set. 2009
2.0	5	Eclair	26 out. 2009
2.0.1	6		03 dez. 2009
2.1	7		12 jan. 2010
2.2	8	Froyo	20 mai. 2010
2.2.1			18 jan. 2011
2.2.2			22 jan. 2011
2.2.3			21 nov. 2011
2.3	9	Gingerbread	06 dez. 2010
2.3.1			dezembro / 2010
2.3.2			janeiro / 2010
2.3.3	10		09 fev. 2011
2.3.4			28 abr. 2011
2.3.5			25 jul. 2011
2.3.6			02 set. 2011
2.3.7			21 set. 2011
3.0	11	Honeycomb	22 fev. 2011
3.1	12		10 mai. 2011
3.2	13		15 jul. 2011
3.2.1			20 set. 2011
3.2.2			30 ago. 2011
3.2.3			01 ago. 2011
3.2.4			15 dez. 2011

3.2.5	13	Honeycomb	janeiro / 2012
3.2.6			15 fev. 2012
4.0	14	Ice Cream Sandwich	18 out. 2011
4.0.1			21 out. 2011
4.0.2			28 nov. 2011
4.0.3	15		16 dez. 2011
4.0.4			29 mar. 2012
4.1	16	Jelly Bean	09 jul. 2012
4.1.1			23 jul. 2012
4.1.2			09 out. 2012
4.2	17		13 nov. 2012
4.2.1			27 nov. 2012
4.2.2			11 fev. 2013
4.3	18		24 jul. 2013
4.3.1			03 out. 2013
4.4	19	KitKat	31 out. 2013
4.4.1			05 dez. 2013
4.4.2			09 dez. 2013
4.4.3			02 jun. 2014
4.4.4			19 jun. 2014
4.4W	20		25 jun. 2014
4.4W.1			06 set. 2014
4.4W.2			21 out. 2014
5.0	21	Lollipop	12 nov. 2014
5.0.1			02 dez. 2014
5.0.2			19 dez. 2014
5.1	22		09 mar. 2015
5.1.1			21 abr. 2015
6.0	23	Marshmallow	05 out. 2015

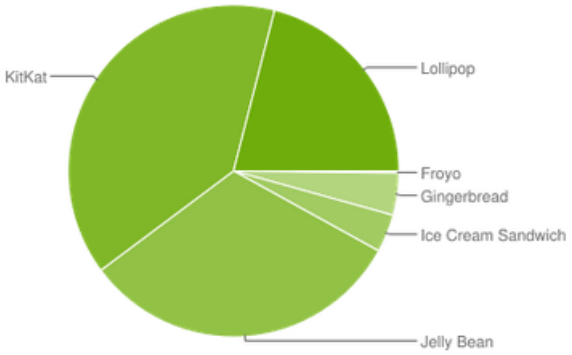
Fonte: ANDROID, 2015 (elaboração nossa)

Uma observação sobre as versões 4.4W, 4.4W.1 e 4.4W.2 é que elas foram desenvolvidas exclusivamente para os novos dispositivos vestíveis (*wearables*), batizados de “Android Wear”. Um dispositivo vestível é um computador que está alocado no espaço pessoal do usuário, controlado pelo usuário, e possui constância de operação e interação, ou seja, está sempre ligado e sempre acessível. Mais notavelmente, ele é um dispositivo que está sempre com o usuário, e permite que o usuário digite comandos ou os execute, enquanto anda ou faz outras atividades (MANN, 1998). Alguns exemplos desses dispositivos vestíveis são os relógios inteligentes (*smartwatch*), óculos de realidade virtual, entre outros.

Além disso, segundo os dados estatísticos do site oficial do Android, percebe-se na figura 3 as versões distribuídas da seguinte maneira:

FIGURA 3 - Gráfico de distribuição das versões do Android

Versão	Codinome	API	Distribuição
2.2	Froyo	8	0.2%
2.3.3 - 2.3.7	Gingerbread	10	4.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	3.7%
4.1.x	Jelly Bean	16	12.1%
4.2.x		17	15.2%
4.3		18	4.5%
4.4	KitKat	19	39.2%
5.0	Lollipop	21	15.9%
5.1		22	5.1%



- Dados coletados em Setembro de 2015.
- Quaisquer versões com distribuição inferior a 0,1% não são mostradas.

Fonte: ANDROID DEVELOPERS, 2015 (tradução nossa)

Segundo a Google (2015), o sistema Android ganha nada mais e nada menos que 1 milhão de novos usuários a cada dia e, atualmente, o Android é utilizado por 1,4 bilhão de pessoas.

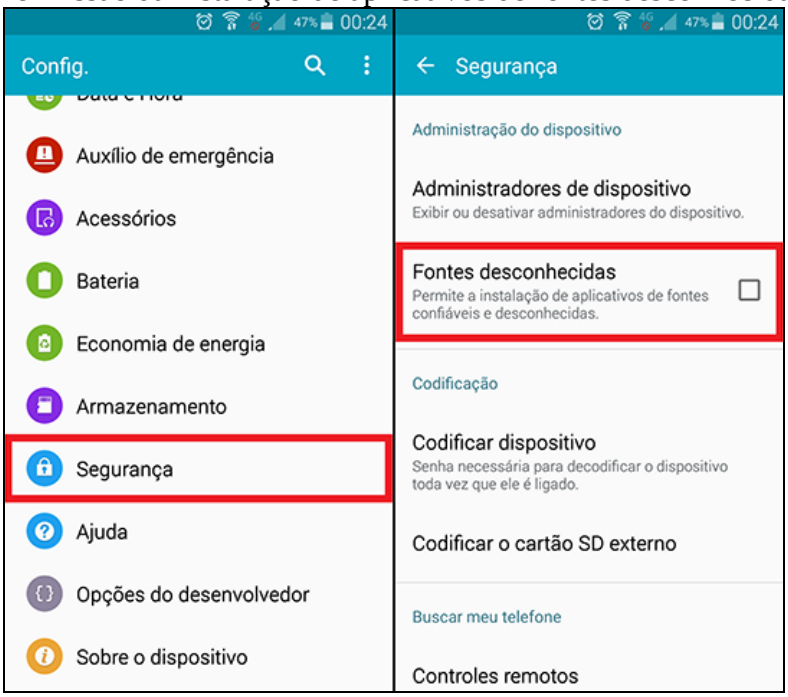
2.7 Google Play Store

De acordo com Kimura (2014), a Google Play Store é uma loja online da Google para distribuição de aplicações, jogos, filmes, música e livros para dispositivos com o sistema Android. Considerada como a loja oficial para os dispositivos Android e anteriormente chamada de “Android Market”, a Google Play Store conta com bilhões de aplicativos de diversos tipos como jogos, redes sociais, mensageiros, corporativo, entretenimento, navegadores, segurança e fotografia, além da venda e aluguel de filmes online e livros digitais. O serviço permite a instalação de aplicativos remotamente, atualizar automaticamente, avaliar e comentar sobre os aplicativos e sugere novos títulos com base nas suas preferências de aplicativos e jogos. Além disso, a Google Play Store permite ao usuário personalizar sua experiência de leitura, localizar e compartilhar livros eletrônicos e assistir aos seus filmes favoritos. A sincronização na nuvem possibilita que o conteúdo esteja disponível na Web e em todos os seus dispositivos Android.

Por se tratar de uma plataforma aberta, os usuários do Android não estão restritos a loja Google Play Store como o único lugar para obter aplicativos. O Android foi construído de

forma a ser prático, e praticidade não combina com restrições e limitações de uso, ou seja, passível de ser “mexido” por qualquer um que tenha conhecimento. É possível instalar aplicativos fora da loja por meio de arquivos APK, que são equivalentes aos arquivos executáveis (EXE) do Windows, por exemplo. Um arquivo APK é um pacote com o aplicativo propriamente dito, cuja instalação fica nas mãos do instalador do sistema (FERREIRA, 2013). Embora o Android permita a instalação de aplicativos de fontes desconhecidas, essa opção vem desativada por questões referentes à segurança e por padrão no Menu de Configurações, como mostra a figura 4.

FIGURA 4 - Permissão da instalação de aplicativos de fontes desconhecidas no Android



Fonte: ANDROID, 2015 (elaboração nossa)

Apesar de seus benefícios e vantagens, a Google Play Store apresenta sérios problemas relacionados à distribuição de aplicativos com códigos maliciosos (TECMUNDO, 2015). Empresas de TI, como a própria Google, estão lidando com essas ameaças constantemente, seja usando scanners para detectar códigos nocivos e bloqueá-los de suas lojas ou retirando programas falsos de sua loja. Dessa maneira, a Google Play Store estaria “supostamente” segura. Segundo a Google (2014), apenas 0,1% de sua lista de aplicativos teria malwares.

Porém, por mais que a Google afirme que a ameaça oferecida por aplicativos maliciosos ao seu sistema operacional seja desprezível, uma pesquisa feita através de um scanner de apps maliciosos desenvolvido por pesquisadores da Universidade de Indiana (2015) mostra que

10% dos aplicativos oferecidos na Google Play Store possuem algum tipo de código malicioso (TECMUNDO, 2015).

3 METODOLOGIA

3.1 Definição de método e metodologia

> Em seu sentido mais geral, método é a ordem que se deve impor aos diferentes processos necessários para atingir certo fim ou um resultado desejado. Nas ciências, entende-se por método o conjunto de processos empregados na investigação e na demonstração da verdade (BERVIAN; CERVO; SILVA, 2002).
>
> Metodologia serve para explicar tudo que foi feito durante um estudo. O objetivo é descrever o método, os participantes, o tipo de pesquisa e os instrumentos utilizados (como entrevistas e questionários), entre outras coisas (MASCARENHAS, 2012).

3.2 Caracterização do estudo

3.2.1 Segundo os objetivos

Segundo objetivos, a pesquisa será desenvolvida de forma qualitativa dissertativa, fazendo o uso de referências bibliográficas, como artigos, livros e notícias sobre as falhas relatadas com maior frequência pelos autores.

3.2.2 Segundo as fontes de dados

Segundo as fontes de dados, a pesquisa realizada no projeto é de natureza bibliográfica, pois será elaborada consultando materiais já publicados, como fóruns, artigos científicos e livros.

3.2.3 Segundo a forma de abordagem

Segundo a forma de abordagem, a pesquisa realizada no projeto é de natureza qualitativa, porque ela é descritiva e os dados obtidos serão analisados indutivamente.

3.2.4 Segundo o procedimento de coleta de dados

Segundo o procedimento da coleta de dados, a pesquisa será caracterizada pela documentação indireta, ou seja, utilização de dados coletados por outras pessoas obtidos por intermédio de pesquisa bibliográfica (fontes secundárias).

3.2.5 Segundo a amostra de coleta de dados

Segundo a amostra da coleta de dados, é caracterizada por tipo de dados, que são a descrição dos critérios da escolha da amostra, apresentando suas características e como os dados serão coletados.

4 RESULTADOS

Com foco em atender ao objetivo geral proposto os resultados são descritos neste capítulo partindo das análises feitas no uso do sistema Android.

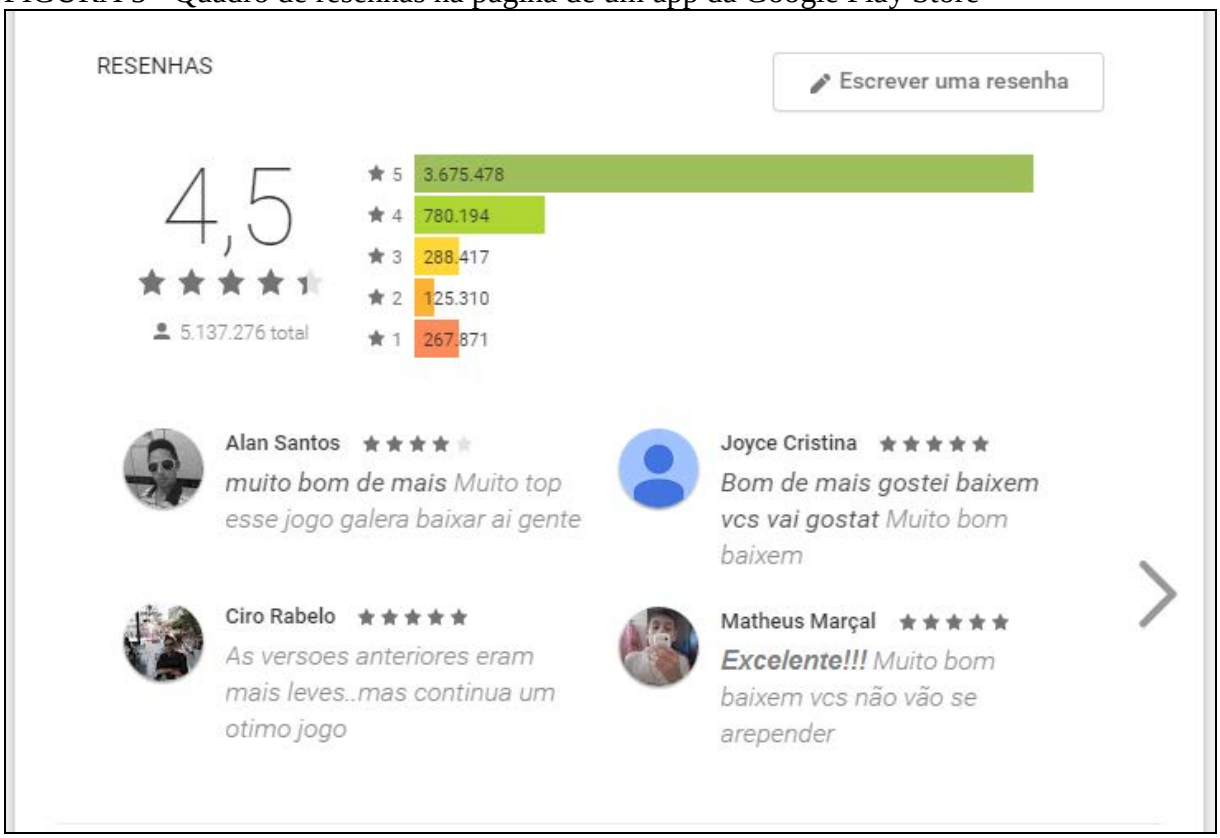
4.1 Autenticidade de um aplicativo da Google Play Store

Todos os dias, novos aplicativos com códigos maliciosos se aproveitam da versatilidade do sistema Android e da loja Google Play Store para roubar dados dos usuários e obter acesso não autorizado ao seu dispositivo. Mas como identificar se tal aplicativo da loja é realmente confiável?

De acordo com TECHTUDO (2013), há algumas características na página de cada aplicativo na loja que permitem concluir se vale a pena instalar no dispositivo ou não. Observando algumas informações dos aplicativos que estão disponibilizados na Google Play Store, é possível reduzir os riscos de se fazer o download de um programa infectado.

O quadro com as resenhas (comentários) dos usuários que fizeram o download do aplicativo, como mostra a figura 5, é um dos melhores recursos da loja. Além de poder observar a média de avaliação, é possível conhecer mais sobre os apps com as descrições, elogios e reclamações de outras pessoas, mantendo os usuários atentos às queixas de problemas ou atividades suspeitas.

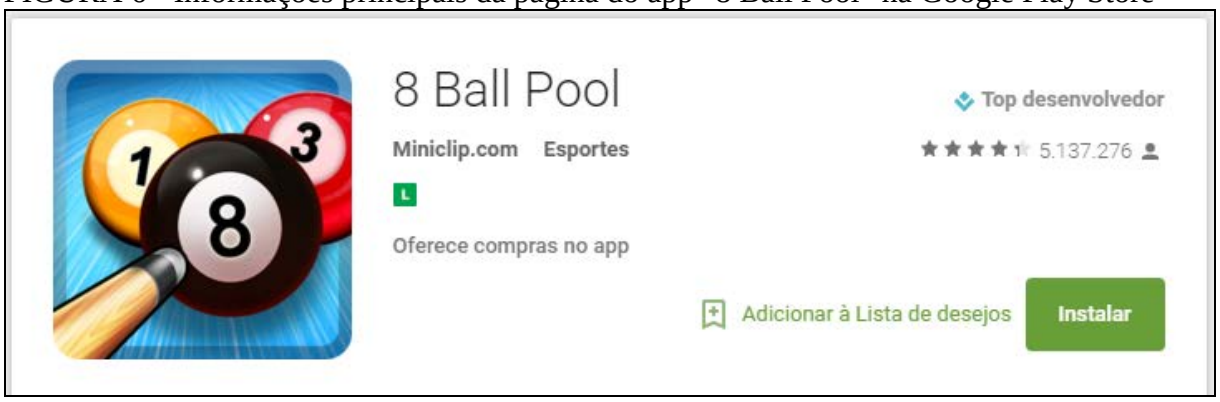
FIGURA 5 - Quadro de resenhas na página de um app da Google Play Store



Fonte: GOOGLE PLAY STORE, 2015

Além desse recurso, ao lado do ícone do aplicativo na Google Play Store, há um número entre parênteses indicando quantas pessoas avaliaram o app. Quanto mais avaliações positivas o programa tiver, maiores serão as chances de ser uma aplicação confiável. Por meio da figura 6, percebe-se que o aplicativo “8 Ball Pool” é bem avaliado e muito procurado pelos usuários.

FIGURA 6 - Informações principais da página do app “8 Ball Pool” na Google Play Store

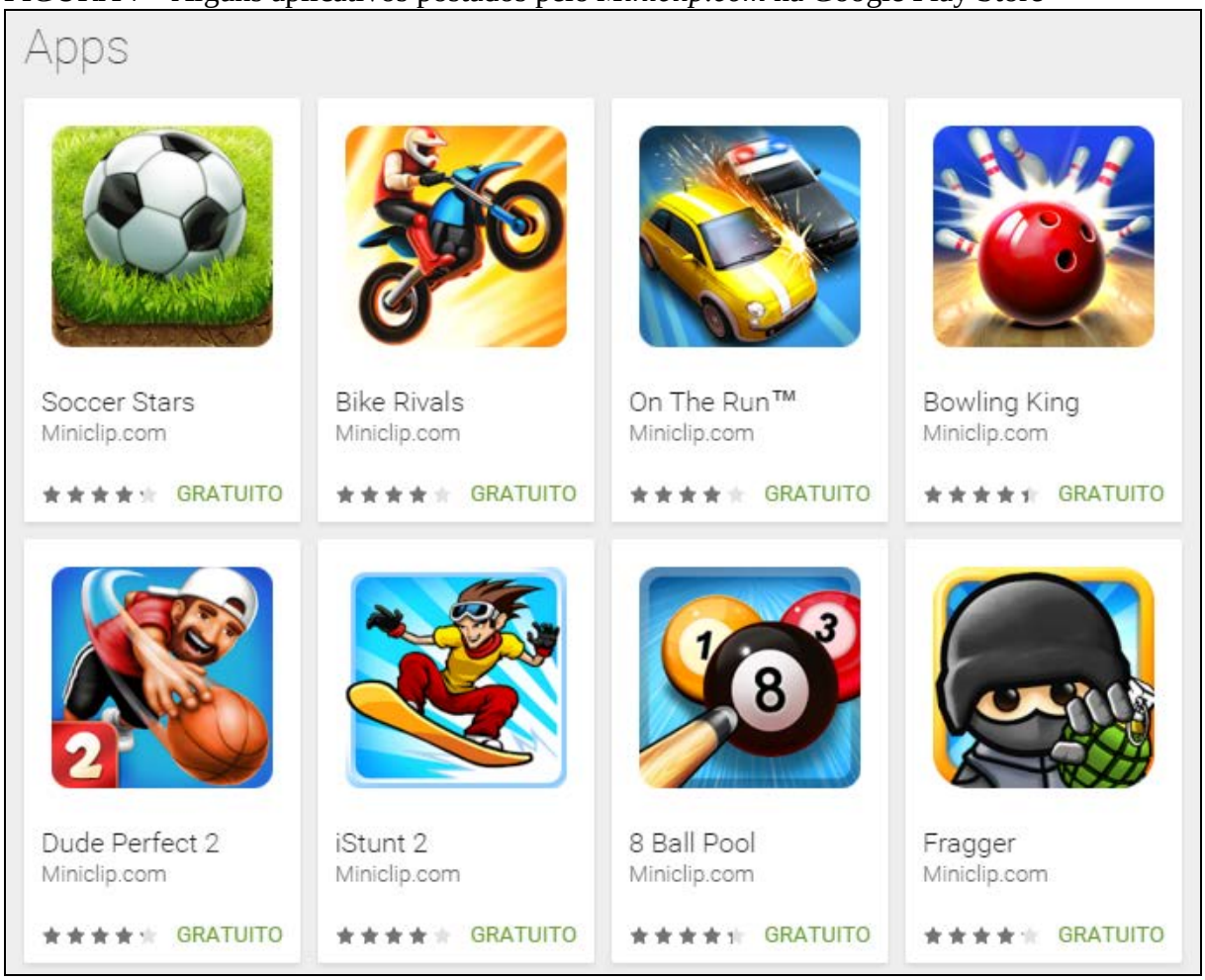


Fonte: GOOGLE PLAY STORE, 2015

Também pela figura 6, compreende-se que o desenvolvedor *Miniclip.com* é classificado como “Top desenvolvedor”. Quanto mais conhecidos forem os desenvolvedores dos apps, maior será a probabilidade deles estarem seguros. É muito provável que desenvolvedores

classificados como “Top desenvolvedor”, que nem o *Miniclip.com*, ofereçam aplicativos seguros, sem malwares. Caso ainda há a desconfiança sobre a procedência dos desenvolvedores, busque informações deles em sites de busca. Também é bom checar na própria página da Google Play Store, como é exibida na figura 7, os outros aplicativos que o desenvolvedor produziu, observando possíveis contradições entre os programas oferecidos, o número de downloads que foram feitos e as resenhas dos usuários.

FIGURA 7 - Alguns aplicativos postados pelo *Miniclip.com* na Google Play Store

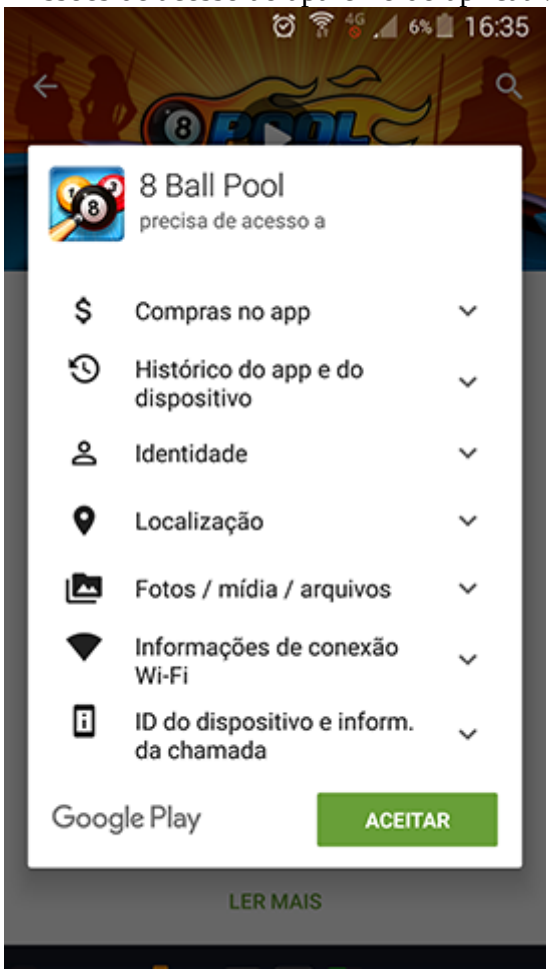


Fonte: GOOGLE PLAY STORE, 2015

Após autorizar a instalação do aplicativo no dispositivo Android, surgirá uma janela que mostra certas permissões de acesso ao seu dispositivo que são necessárias para o funcionamento do aplicativo, como é exibido na figura 8. A partir disso, o app será instalado somente se o usuário aceitá-las (exceto no Android 6.0, porque a Google mudou essa funcionalidade). É sempre bom analisar se o programa terá acesso às funções que condizem com as suas atividades, desconfiar de tais acessos que excedam a sua finalidade ou tenham

contradições, como apps de *wallpapers* (papéis de parede) que utilizam a sua agenda de contatos ou jogos que controlem a câmera do aparelho.

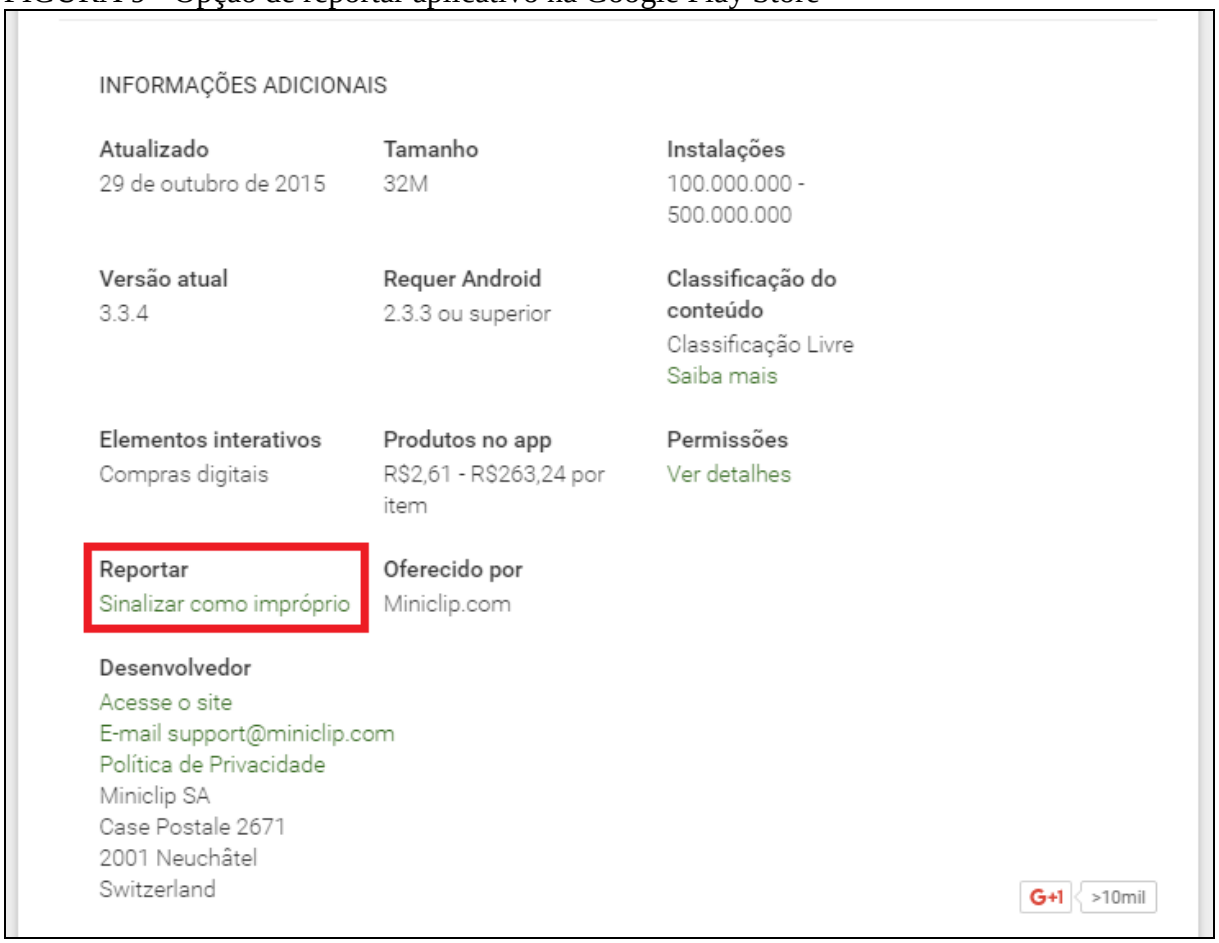
FIGURA 8 - Permissões de acesso ao aparelho do aplicativo “8 Ball Pool”



Fonte: GOOGLE PLAY STORE, 2015

Finalmente, não deixe de reportar para a Google Play Store sobre aplicativos suspeitos. No final da página do aplicativo na loja, tem as informações adicionais do mesmo. Depois, selecione a opção “sinalizar como impróprio”, como é devidamente mostrada na figura 9. Dessa maneira, a equipe de segurança da Google, de acordo com a quantidade de denúncias, poderá checar ameaças e excluí-las.

FIGURA 9 - Opção de reportar aplicativo na Google Play Store



Fonte: GOOGLE PLAY STORE, 2015

Seguindo essas dicas, a probabilidade de baixar uma aplicação maliciosa é muito menor. Muitos dos aplicativos maliciosos são facilmente detectados seguindo uma ou duas dicas dessa lista. Dentre os casos mais comuns, evitem aplicações com nomes famosos feitos por desenvolvedoras desconhecidas. Exemplos mais comuns são nomes de títulos clássicos, como “Age of Empires” e “Counter-Strike”, que não possuem versões oficiais na Google Play Store. Também é preciso, a qualquer custo, evitar a instalação de programas desconhecidos fora da loja, assim como o uso de apps que alterem funcionalidades de outros apps (TECHTUDO, 2013). Com essa prática de usabilidade, o usuário estará se protegendo ainda mais contra a instalação de aplicativos com códigos maliciosos.

4.2 Funcionamento das atualizações de versões do Android

Como forma de manutenção das atualizações do sistema, a Google disponibiliza o código fonte atualizado as fabricantes, a qual se responsabiliza pela distribuição com suas personalizações. Com tais atualizações, são corrigidas vulnerabilidades encontradas, *bugs* e

melhorias de desempenho (VIDAS, 2011). Com as informações da tabela 2, percebe-se o ciclo de como são feitas correções no sistema Android.

TABELA 2 - Ciclo de produção das atualizações do Android

Passo	Descrição
1	Vulnerabilidade é descoberta
2	Vulnerabilidade é reportada via NDA (*Non-Disclosure Agreement*)
3	Se a vulnerabilidade for proveniente do código fonte do sistema, o erro é corrigido pela equipe de desenvolvimento do Android
4	Fabricantes recebem o código fonte concertado e adequam as suas personalizações
5	Operadoras de telefonia também se adequam ao código e as suas personalizações
6	Os usuários recebem a atualização via OTA e instalam o patch de segurança

Fonte: ROTONDO, 2015, p. 4 (adaptação nossa)

Segundo Rotondo (2015), a complexidade deste ciclo deve ser considerada, porque o sistema é modificado em vários pontos, começando pela equipe de desenvolvimento do Android, passando por modificações do fabricante e posteriormente da operadora de telefonia. E não para por aí! A complexidade aumenta ainda mais devido à inúmera quantidade de fabricantes, dispositivos Android e operadoras diferentes no mercado. A parte mais simples seria apenas o recebimento de atualização da versão via OTA, isto é, *Over The Air* (“Pelo Ar”).

A atualização via OTA é um método bastante utilizado pelos dispositivos Android para atualizar a versão do sistema operacional. Como o próprio nome diz, o usuário recebe a atualização pela Internet. Se o aparelho possui alguma atualização disponível, ela simplesmente surge como uma notificação na tela. O usuário não é obrigado a instalar a nova versão, porém a Google recomenda que esse processo seja feito para garantir a segurança, já que a mesma inclui correções de vulnerabilidades e *bugs* e melhorias de recursos. Além disso, a grande vantagem para esse método de atualização é que não é necessário fazer algum tipo de backup do dispositivo.

Atualmente, um dos problemas enfrentados é a falta de atualizações para todos os dispositivos Android simultaneamente. Muitos aparelhos datados como ultrapassados não chegam a receber tais melhorias, permanecendo vulneráveis e forçando o usuário a comprar um novo dispositivo com a versão mais atualizada. A alternativa para esses usuários é a técnica de *flashing*, que consiste em adquirir e instalar uma versão oficial modificada do sistema operacional Android (ANDROID DEVELOPERS, 2015).

Uma dessas distribuições mais usadas é fornecida pela empresa CyanogenMod, que em 2012 já contava com mais de 1 milhão de distribuições instaladas (GRAVEHEART, 2012), e

possibilita aparelhos mais antigos usarem as versões mais recentes do Android, como a versão 6.0 com total compatibilidade. Porém, todo cuidado é pouco para esse tipo de prática. Não há a absoluta certeza de que todas as funcionalidades e recursos que o aparelho oferece tenha total compatibilidade com a instalação de uma versão customizada.

4.3 Root nos dispositivos Android

Quem tem experiência no Linux, o termo *root* já é familiar, ainda mais que o Android foi baseado no *Kernel* do Linux 2.6. Fazer o *root* no dispositivo significa se tornar um super-administrador do sistema, ou seja, ter acesso às partes do Android que são inacessíveis para um usuário comum (OLHAR DIGITAL, 2014).

Dessa maneira, é possível fazer alterações no sistema operacional que antes eram impossíveis, como alterar consumo de bateria, modificar o *clock* do processador, apagar aplicativos fixos no sistema, como os *bloatwares*[3] embutidos no sistema pela fabricante e/ou pela operadora (GIZMODO BRASIL, 2012).

No entanto, os arquivos importantes do sistema ficam inteiramente expostos e eles podem ser apagados acidentalmente, o que pode causar a inutilização do aparelho. O quadro fica ainda pior com a presença de um aplicativo malicioso instalado no sistema Android, principalmente se ele tiver a permissão de acesso ao gerenciador de arquivos, ou até mesmo se o próprio app controlador do *root* for um malware. O processo de realização do *root* também não é imune a falhas, logo, o aparelho simplesmente se transforma num “tijolo inútil” (OFICINA DA NET, 2013).

Além disso, a modificação do sistema operacional é uma violação do contrato de garantia com fabricantes e operadoras. Embora isso não seja crime, a ação anulará qualquer possibilidade de reparo, a menos que ela seja desfeita (LIFEHACKER, 2013). Depois de “rootear” o aparelho, dependendo do app controlador do *root* instalado, ele pode possuir a função de remover a permissão do super-administrador, retornando ao estado de um usuário comum. Outra desvantagem de fazer o *root* no dispositivo Android é que o usuário deixará de receber atualização via OTA (ROCHA, 2013).

[3] Programas pré-instalados que provê funcionalidade mínima, requerendo quantidades desproporcionais de armazenamento, memória e processamento.

4.4 Introdução sobre as versões 4.3 até 5.1 do Android

Com o intuito de entender melhor as versões do Android estudadas, neste capítulo estão descritas algumas características delas, como novas funcionalidades e melhorias de recursos.

Um detalhe importante sobre as versões 4.3 e inferiores é que, em janeiro de 2015, a Google tomou uma decisão que prejudicou cerca de 930 milhões de usuários do Android 4.3 e anteriores e os colocaram numa posição frágil no que diz respeito a sua segurança. A companhia simplesmente decidiu parar de lançar atualizações de segurança do módulo *WebView* para essas versões, abandonando todas as pessoas que não estão usando as edições mais recentes do sistema Android, no caso, 4.4 *KitKat*, 5.0 *Lollipop* e 6.0 *Marshmallow* (CANALTECH, 2015).

O módulo *WebView* é usado com bastante frequência no desenvolvimento de aplicativos para o Android. Esse componente é capaz de exibir páginas da web sem que seja necessário abrir um programa em separado. Essa característica é constantemente usada por pessoas mal intencionadas para fazer acessos não autorizados aos smartphones e roubar discretamente informações pessoais. Com o cancelamento dessas atualizações, os crackers poderão “deitar e rolar” nos dispositivos Android equipados com a versão 4.3 ou inferiores (WINDOWSTEAM, 2015).

Segundo o chefe de segurança do Android, Adrian Ludwig, que emitiu um comunicado através das redes sociais[4] explicando o verdadeiro posicionamento da empresa a respeito desse desligamento, o Android com suas 5 milhões de linhas de código torna inviável a solução desse problema por exigir alterações em porções significativas do código e não ser mais prático fazer isso com segurança (TECMUNDO, 2015). Isso porque a partir da versão 4.4, o *WebView* também foi substituído pelo plugin derivado do “Chromium Project”. Quando a empresa de segurança Rapid7 revelou uma falha crítica envolvendo a antiga ferramenta, a resposta do Google não poderia ser menos problemática: o *WebView* não será mais atualizado por ter sido substituído por um padrão mais robusto (MEIO BIT, 2015).

Para contornar esse problema, Adrian Ludwig deu sugestões preciosas para todos os usuários dessas versões que foram abandonadas pela Google. Uma delas, basicamente é usar navegadores que não usem o *WebView*, mas são constantemente atualizados com pacotes de

[4] https://plus.google.com/+AdrianLudwig/posts/1md7ruEwBLF

segurança. O Chrome, da própria Google, e o Firefox, da Mozilla, são as duas alternativas sugeridas pelo chefe de segurança do Android. Outra sugestão para aqueles que programam para o Android, Ludwig pede que os desenvolvedores garantam que apenas páginas confiáveis usem o módulo *WebView* em suas aplicações. Além disso, fornecer um método próprio para renderização de páginas para versões anteriores a 4.3 do SO também pode evitar mais dores de cabeça para os usuários. Outras dicas de segurança para proteger aplicativos podem ser encontradas no site Android Developers (TECMUNDO, 2015).

4.4.1 Android 4.3 (API 18)

A versão 4.3 do Android, conhecida como a última versão do *Jelly Bean*, foi lançada em julho de 2013. Ela trouxe melhorias e novas funcionalidades aos usuários e desenvolvedores em relação às versões anteriores, como o Android 4.1 e 4.2.

Uma das principais mudanças do Android 4.3 é que a partir dessa versão, passam a existir perfis restritos no sistema Android, voltados aos pais que compartilham seus smartphones e tablets com os filhos. Novos recursos permitem limitar o acesso das crianças dentro dos aplicativos, por exemplo, para elas não comprarem jogos e itens sem o consentimento dos responsáveis e, obviamente, restringir coisas mais abrangentes, como a navegação, proibindo o acesso de conteúdos pornográficos ou determinadas redes sociais (OLHAR DIGITAL, 2013).

Outro novo recurso suportado é o “Bluetooth Smart”, que conecta dispositivos Android economizando o consumo de bateria, e também o AVRCP (Perfil de Controle Remoto de Áudio e Vídeo), que é um perfil Bluetooth para controlar a reprodução de mídia em dispositivos remotos, como Smart TVs, Set-top boxes, entre outros (TECHTUDO, 2013).

Além dessas novas funcionalidades, o Android 4.3 também é compatível com o OpenGL ES 3.0. Isso trouxe uma evolução na renderização dos gráficos - e tudo em tempo real na definição nativa de 1080p (*Full HD*) - que é uma ótima novidade para os desenvolvedores de games para Android (TECNOBLOG, 2013).

O teclado do sistema Android também foi melhorado na versão 4.3. O algoritmo dele foi refeito, que promete fazer o sistema reconhecer mais rapidamente o que usuário digita e prever as palavras que o usuário quer escrever. Junto com essa melhoria, a ferramenta de

autocompletar foi colocada no aplicativo de chamadas, que reconhece o número que o usuário está digitando para sugerir os telefones relacionados (CNET, 2013).

4.4.2 Android 4.4 (API 19)

Lançado em outubro de 2013, a versão 4.4, denominada *KitKat*, trouxe o Android "para todos". Mas como para todos? Essa versão foi projetada para o sistema trabalhar com extrema rapidez, oferecer ótima usabilidade e total compatibilidade com o dispositivo, mesmo que ele tenha apenas 512 MB de RAM (ANDROID DEVELOPERS, 2013). Não é atoa que o Android 4.4 é, até o momento, a versão mais utilizada entre todas as versões já produzidas pela Google[5]. Além de trazer novas funções, a versão 4.4 também possui mudanças marcantes no design.

Com o *KitKat*, a Google apostou no branco como cor de destaque, o que se vê de imediato nos símbolos da tela inicial e na barra de notificações. O "rastro" do teclado ao se deslizar o dedo pelas teclas e muitos outros elementos também aparecem na cor branca. A Google justifica essa mudança alegando que assim a atenção se voltará mais para o conteúdo, e não mais para o sistema operacional em si (TECHTUDO, 2013).

Outra novidade diz respeito aos aplicativos. Os desenvolvedores ganharam a possibilidade de mudar as cores e ícones dos seus apps com versões próprias, respeitando algumas diretrizes do Android 4.4 e da linha geral de design do Google. O ícone de partilha de conteúdo, por exemplo, não pode ser substituído por um símbolo completamente diferente, mas pode ter outras cores (ANDROIDPIT, 2013).

O Android 4.4 também introduziu dois modos de tela inteira. *Lean Back* e *Immersive*. No modo *Lean Back*, todas as barras de sistema desaparecem completamente, mas podem ser acessadas de novo através de um toque na tela. Ele é ideal para a reprodução de vídeos. Já o modo *Immersive* também faz desaparecer todas as barras de sistema, e para fazê-las aparecer de novo é necessário deslizar o dedo de cima para baixo na tela. Esse é o modo ideal para jogos e outros aplicativos de multimídia (ANDROID DEVELOPERS, 2013).

Além disso, a Google percebeu a necessidade de adicionar mais interações via gestos do usuário com o dispositivo a partir dessa versão 4.4. Nos smartphones com grandes telas, o controle com uma só mão fica cada vez mais difícil. Para contornar esse problema, a Google

[5] http://developer.android.com/intl/pt-br/about/dashboards/index.html

criou dois novos gestos para o polegar: um clique duplo, que aumenta o campo automaticamente, e um clique duplo seguido do deslizar do dedo para cima ou para baixo, que aumenta ou diminui gradualmente o campo de visão da tela, como a visualização de um mapa no aplicativo Google Maps (ANDROIDBEAT, 2013).

4.4.3 Android 5.0 (API 21)

Lançado em novembro de 2014, o Android 5.0, batizado como *Lollipop*, foi um *release* focado na interface de usuário, usabilidade, animações e experiência do usuário.

Foi nesta versão que surgiu o “Material Design”, um guia completo sobre como implementar o visual, animações e interação entre componentes de um layout, levando em consideração que o Android se tornou uma plataforma comum para vários dispositivos, como smartphones, tablets, dispositivos vestíveis (Android Wear), óculos (Google Glass), TVs (Android TV) e carros (Android Auto). Isso é o mais importante, uma vez que as técnicas do “Material Design” não precisam ser implementadas somente nos smartphones e tablets, porque a Google criou um padrão de design consistente em várias plataformas que foram citadas anteriormente (ANDROID DEVELOPERS, 2014).

Dentre outras melhorias, as notificações agora também aparecem na tela de bloqueio (*Lock Screen*), e as Head-ups (*head-up notifications)*, que aparecem no topo da tela com alta prioridade. Um exemplo de Head-ups é a ligação que permite atender ou rejeitar uma ligação telefônica diretamente na notificação. Antigamente, esse recurso não existia e a aplicação da ligação mostrava uma tela cheia para o usuário decidir se atende ou não a ligação (TECNOLOGIA E TAL, 2014).

Outra novidade interessante foi o projeto “Volta”, que trouxe ferramentas para auxiliar a análise do uso da bateria nos aplicativos. Também foi modificada a tela de aplicativos recentes (*Overview Screen*), que mostra as últimas tarefas que estão sendo executadas, sendo que um aplicativo pode conter uma ou mais tarefas. Além disso, o Android 5.0 também suporta a OpenGL ES 3.1, trazendo um desempenho superior nos jogos 2D e 3D (LECHETA, 2015, p. 39).

4.4.4 Android 5.1 (API 22)

Quatro meses depois da distribuição do Android 5.0, a Google lançou uma atualização para a mesma em março de 2015, a versão 5.1. A atualização corrige vulnerabilidades e *bugs*, otimiza o desempenho do software, melhorando a estabilidade e oferece novas funcionalidades.

O Android 5.1 traz suporte nativo ao gerenciamento de múltiplos cartões SIM, em caso de dispositivos que possuem dois ou mais entradas para o mesmo. A nova função que traz mais segurança ao Android se chama “Proteção de Dispositivos”, pela qual smartphones ou tablets permanecem bloqueados até que o usuário entre com sua conta Google. A função continua ativa mesmo quando o dispositivo é redefinido para as configurações de fábrica, e a função de confirmação da conta permanecerá ativa. Este novo recurso de segurança estará presente em todos os dispositivos que rodem com Android 5.1, independentemente da fabricante (ANDROIDPIT, 2015).

O recurso de chamada por voz em alta definição está presente no Android 5.1 e vai além de um equalizador para chamadas. A nova ferramenta reduz ainda mais o ruído ambiente, fazendo com que a chamada mantenha a voz mais cristalina na percepção do ouvinte (CANALTECH, 2015).

A estabilidade do sistema também foi otimizada como um todo e isso refletirá na drenagem de carga na bateria. As melhorias também incluem mais rapidez em conexões sem fio, como Wi-Fi ou dispositivos emparelhados via Bluetooth, que terão atalhos rápidos acessíveis ao usuário (TALKDROID, 2015).

4.4.5 O anúncio do Android 6.0 (API 23) e suas expectativas

Inicialmente, no evento “Google I/O 2015” foi anunciado o Android M, que é a versão prévia para desenvolvedores do novo Android 6.0. A letra “M” dá sequência à letra “L” de *Lollipop*, e a Google está seguindo essa nomenclatura para cada versão lançada (TECHTUDO, 2015).

O lançamento oficial da versão 6.0 do Android ocorreu em outubro de 2015 e foi batizado como *Marshmallow*. Atualmente, ela é a versão mais recente que a Google lançou (ANDROID, 2015).

Dentre as melhorias anunciadas no Android 6.0 estão o leitor de impressão digital e o “Android Pay”, uma plataforma aberta para fazer pagamentos por meio de cartões, Internet e NFC (LECHETA, 2015, p. 40).

Uma das novas funcionalidades bem interessantes é o novo sistema de controle de permissões dos aplicativos, o qual possibilitará que o usuário conceda a permissão individualmente. Por exemplo, será possível optar por dar acesso à câmera enquanto nega-se a permissão para acessar o GPS (MOBILEXPERT, 2015).

Outra funcionalidade atraente do Android 6.0 são os *App Links*, que permitem que determinados aplicativos sejam escolhidos como padrão ao abrir links de determinado domínio. Exemplos clássicos são os apps de redes sociais, como o Facebook, Twitter, Google Plus, entre outros. Nesse caso, qualquer link desses domínios pode ser aberto diretamente em seus respectivos aplicativos sem a necessidade de perguntar ao usuário qual aplicativo deve ser escolhido (ANDROIDPIT, 2015).

O “Google Now” também recebeu atualizações e agora ele pode ser chamado diretamente da tela de bloqueio, pela opção no canto esquerdo inferior da tela (TECNOBLOG, 2015).

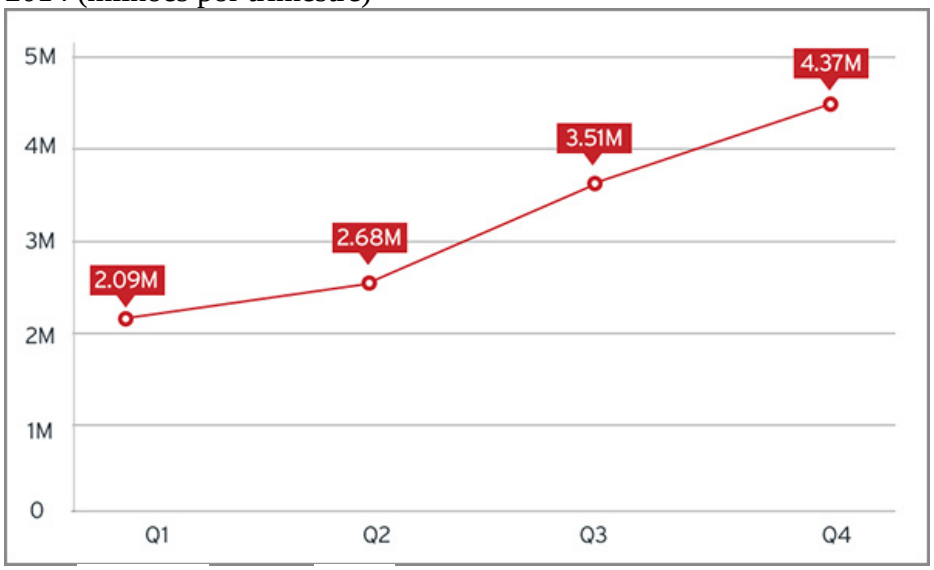
Enfim, muitas alterações foram realizadas no *Kernel* do sistema, mas não são tão “notáveis” pelos usuários. Contudo, as implementações de segurança, como o armazenamento criptografado e o backup mais completo, são importantes num quadro geral, em que existem muitos dispositivos Android que estão mais vulneráveis (TECMUNDO, 2015).

4.5 As principais vulnerabilidades do Android

As vulnerabilidades dos sistemas móveis sempre foram assunto em pauta desde o momento em que o uso destes dispositivos se difundiu pelo mundo. Segundo a Kaspersky (2014), o primeiro worm descoberto que poderia afetar sistemas *mobile* foi o “Cabir”. Apesar de não possuir muitas funcionalidades para causar danos, ele fez história ao deixar claro que um sistema *mobile* poderia ser afetado por esse tipo de malware. O “Gingermaster”, por exemplo, foi um *Trojan* famoso por infectar dispositivos Android, mais especificamente a versão Android 2.3.3 da plataforma (SCHWARTZ, 2011).

De acordo com a figura 10, a quantidade de aplicativos maliciosos criados e vulnerabilidades exploradas não param de aumentar, como já poderia se esperar.

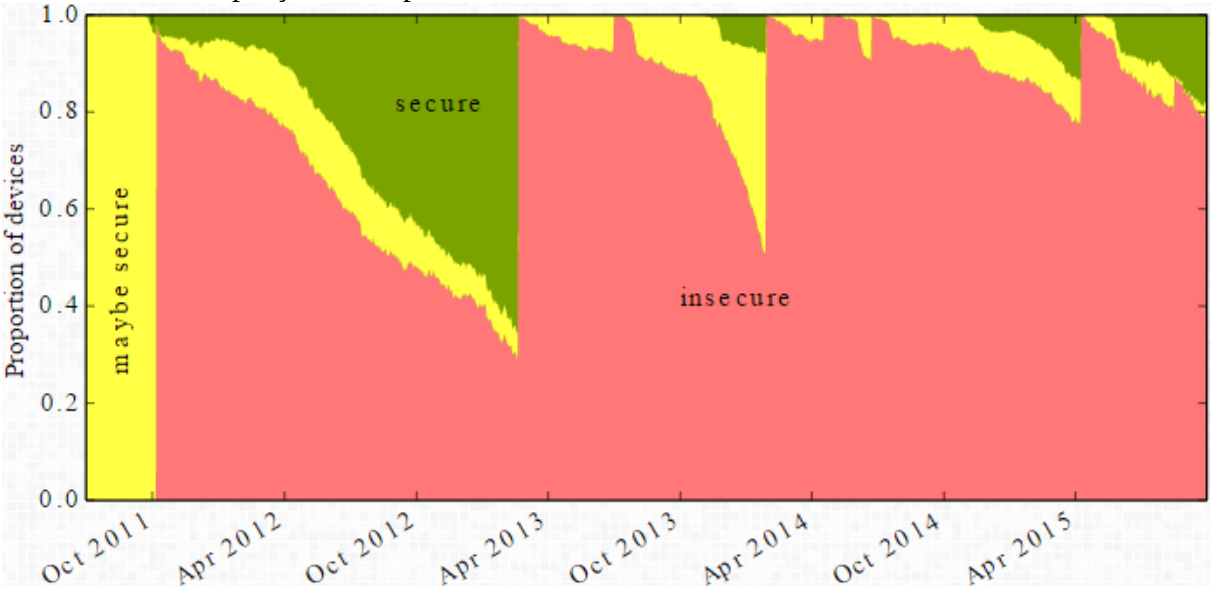
FIGURA 10 - Crescimento de aplicativos maliciosos e de alto risco em 2014 (milhões por trimestre)



Fonte: TREND MICRO, 2014

Dentre todos os sistemas móveis, o Android é o mais afetado. Recentemente, pesquisadores da Universidade de Cambridge (2015), no Reino Unido, confirmaram num estudo sobre vulnerabilidades um fato que tem sido sempre o grande problema quando se trata de dispositivos Android: 87% desses dispositivos estão expostos a, no mínimo, uma vulnerabilidade crítica. Os dados para o estudo vieram em mais de 20.000 dispositivos Android com o aplicativo “Device Analyzer” instalado, e foram testados contra 13 erros conhecidos que têm sido no domínio público nos últimos 5 anos (SILICON ANGLE, 2015).

FIGURA 11 - Proporção de dispositivos rodando versões vulneráveis do Android



Fonte: ANDROID VULNERABILITIES, 2015

Por meio da figura 11, é possível perceber a estimativa da proporção de dispositivos Android rodando versões rotuladas como “inseguras”, “talvez inseguras” e “seguras”. Além disso, a tabela 3 lista todas as vulnerabilidades documentadas pela Android Vulnerabilities.

Conforme a Android Vulnerabilities (2015), para rotular um dispositivo como “inseguro” ele deve atender às duas condições: primeiramente, o dispositivo está executando uma versão do Android que é vulnerável a pelo menos uma das vulnerabilidades listadas. Segundamente, o dispositivo não recebeu uma atualização que poderia corrigir a vulnerabilidade.

Da mesma forma, um dispositivo é rotulado como “talvez inseguro” quando ele estiver executando uma versão do Android que é vulnerável a pelo menos uma das vulnerabilidades listadas e o dispositivo recebeu uma atualização que poderia corrigir a vulnerabilidade.

Finalmente, rotular um dispositivo como “seguro” significa que ele está executando uma versão do Android que não é vulnerável a qualquer uma das vulnerabilidades listadas.

TABELA 3 - Lista de Vulnerabilidades documentadas pela Android Vulnerabilities

Nome da Vulnerabilidade	Data da descoberta
KillingInTheNameOf psneuter ashmem	13 jul. 2010
exploid udev	15 jul. 2010
levitator	10 mar. 2011
Gingerbreak	18 abr. 2011
zergRush	06 out. 2011
APK duplicate file	18 fev. 2013
APK unchecked name	30 jun. 2013
APK unsigned shorts	03 jul. 2013
Fake ID	17 abr. 2014
TowelRoot	03 mai. 2014
ObjectInputStream deserializable	22 jun. 2014
Stagefright	08 abr. 2015
Stagefright2	15 ago. 2015

Fonte: ANDROID VULNERABILITIES, 2015

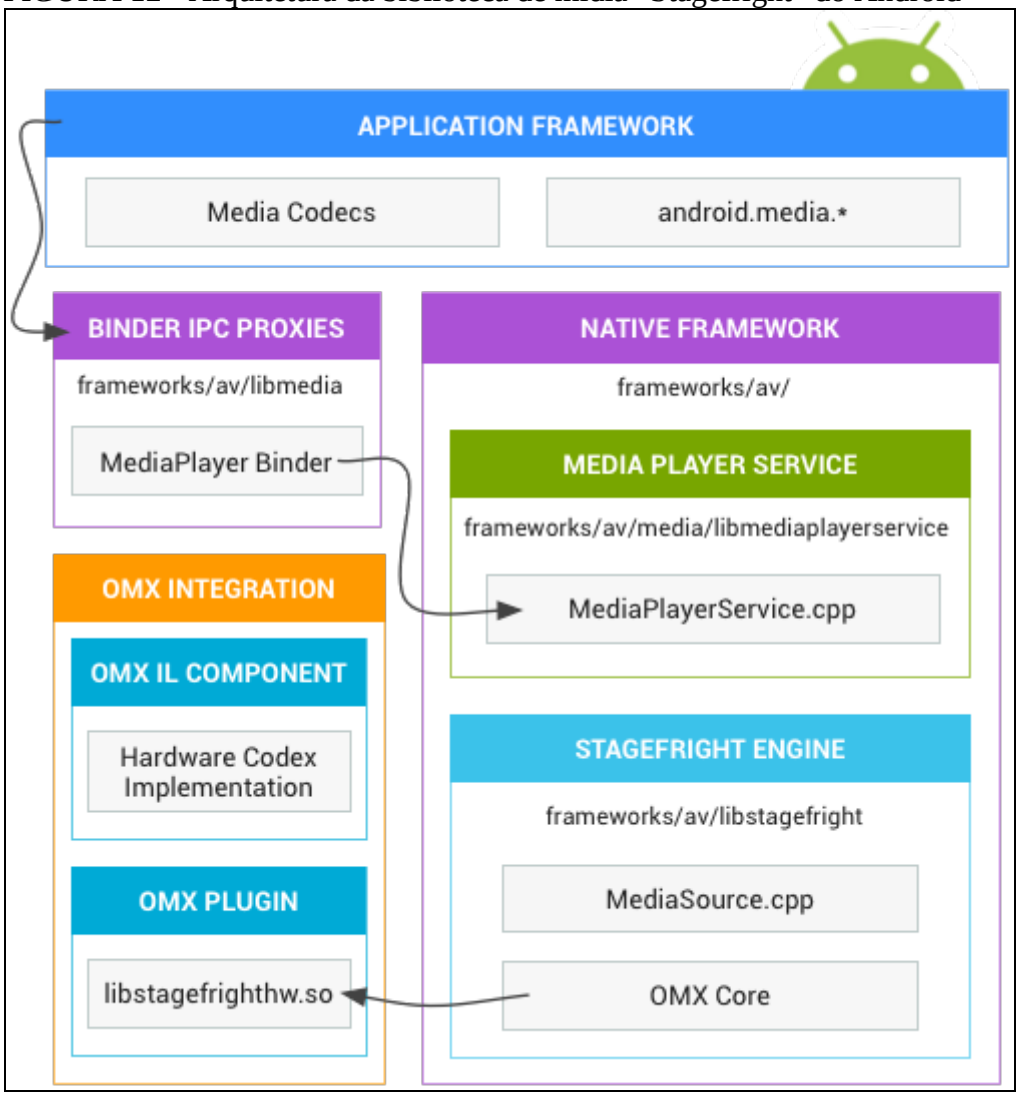
Dentre as 13 vulnerabilidades, serão discutidas quatro delas, que são mais importantes e, por consequência, mais difundidos pela mídia e afetam os dispositivos com versões mais recentes do Android acima da 4.3.

4.5.1 Stagefright 1.0

Construído sobre dezenas de gigabytes de código fonte do projeto *Android Open Source Project* (AOSP), o sistema *mobile* líder de mercado para smartphones carrega um código

assustador. Nomeada “Stagefright” (AOSP, 2015) é uma biblioteca de mídia que processa vários formatos populares. Devido a este tipo de processamento ser muitas vezes limitado por tempo, a biblioteca é implementada em C++ que é mais propenso à corrupção de memória do que as linguagens *memory-safe*, como Java.

FIGURA 12 - Arquitetura da biblioteca de mídia “Stagefright” do Android



Fonte: ANDROID OPEN SOURCE PROJECT, 2015

Um pesquisador e funcionário da empresa Zimperium zLabs, especialista em soluções envolvendo segurança *mobile*, Joshua J. Drake aprofundou dentro dos “cantos mais profundos” do código fonte do sistema Android e descobriu o que acredita ser uma das piores vulnerabilidades do sistema descobertas até hoje. Estes problemas no código “Stagefright” expõem 95% dos dispositivos Android (FORBES, 2015).

Apresentada nos eventos “Black Hat USA 2015” no dia 05 de agosto de 2015 e “DEF CON 23” no dia 07 de agosto de 2015, a pesquisa de Drake encontrou várias vulnerabilidades de

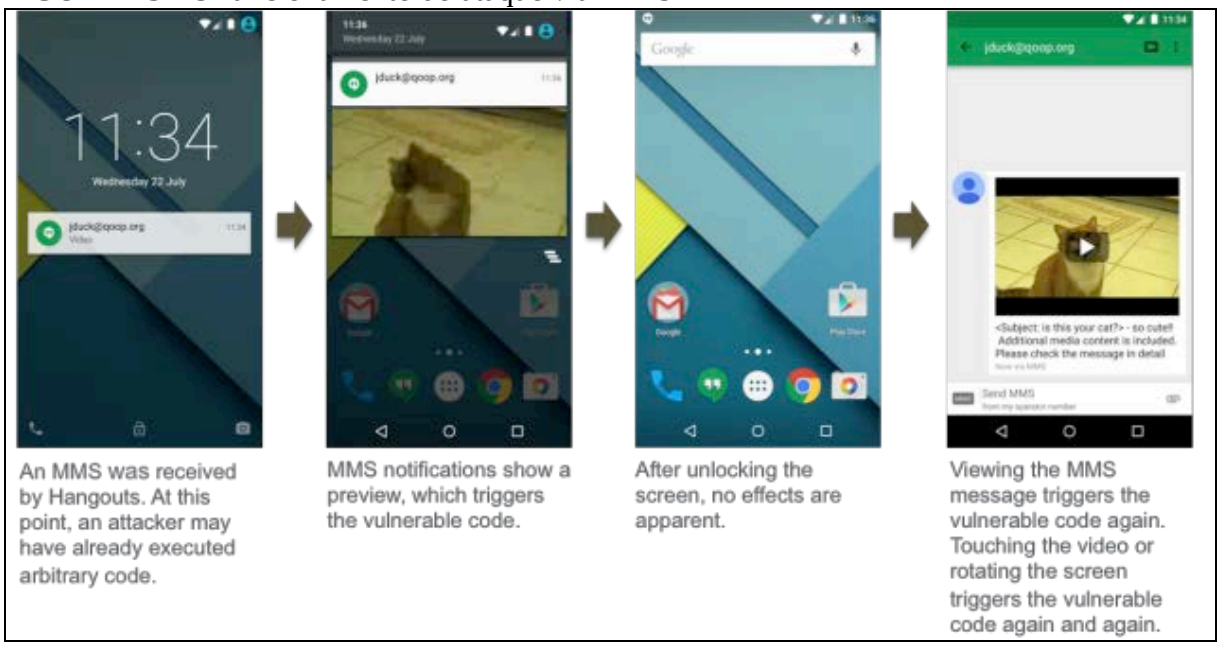
execução remota de código que podem ser exploradas usando vários métodos, sendo que o pior dos quais não requer interação do usuário (FORBES, 2015).

Os invasores só precisam de um número de celular, e utilizando este número eles podem executar o código remotamente através de um arquivo de mídia especialmente criado, entregue via mensagens multimídia (MMS). Um ataque bem sucedido poderia até apagar a mensagem antes mesmo do usuário visualizá-la, sendo que neste caso apenas a notificação seria vista. Essas vulnerabilidades são extremamente perigosas, porque elas não exigem que a vítima tome qualquer ação para ser explorada e não deixa rastros de invasão. Ao contrário de *Spear-Phishing*, em que a vítima precisa abrir um arquivo ou um link enviado pelo atacante, esta vulnerabilidade pode ser acionada enquanto a pessoa dorme, e antes de acordar, o atacante irá remover quaisquer sinais do dispositivo que demonstram que este está comprometido e o dono irá continuar o seu dia como de costume, mas com um telefone infectado (FORBES, 2015).

Segundo a ARS TECHNICA (2015), ataques bem sucedidos podem, no mínimo, fornecer acesso direto ao áudio de um telefone e filmagens da câmera para o armazenamento externo. Aparelhos mais antigos, por concederem privilégios elevados ao sistema de código “Stagefright”, podem permitir que os hackers obtenham o acesso a muito mais recursos.

A figura 13 mostra o passo a passo do ataque via MMS na versão Android 5.1.1 *Lollipop*.

FIGURA 13 - O funcionamento do ataque via MMS



Fonte: ARS TECHNICA, 2015

A primeira tela da figura 13 mostra a mensagem MMS sendo recebida pelo dispositivo móvel da vítima via Google Hangouts. A segunda mostra a notificação da mensagem, o que já ativa o código malicioso. A terceira tela mostra que não existe efeito aparente ao desbloquear a tela do aparelho. E a última mostra a mensagem sendo vista, ativando o código malicioso novamente e continuará ativado a cada ação do usuário, como rotacionar a tela, tocar o vídeo, entre outros (ARS TECHNICA, 2015).

Outros métodos de ataque incluem meios *client-side* (navegadores de Internet, downloads, e-mails), meios fisicamente próximos (NFC, Bluetooth, VCards), meios físicos do dispositivo (cartões SD, *USB On-The-Go*, *Media Transfer Protocol USB*, *Picture Transfer Protocol*) e, possivelmente, outros ainda não identificados (ARS TECHNICA, 2015).

Os dispositivos Android que utilizam a versão 2.2 *Froyo* e superiores estão vulneráveis. E aqueles que têm a versão entre 2.2 *Froyo* e 4.1 *Jelly Bean* se encontram na faixa de pior risco, devido à dificuldade de remoção de problemas como esse (ARS TECHNICA, 2015).

Como pode perceber pela figura 14, muitos fabricantes foram afetados por essa vulnerabilidade.

FIGURA 14 - Lista de Fabricantes afetados pela vulnerabilidade do Stagefright

Vendor Information (Learn More)

Vendor	Status	Date Notified	Date Updated
Amazon	Affected	-	28 Jul 2015
Barnes and Noble	Affected	-	28 Jul 2015
Google	Affected	-	28 Jul 2015
HTC	Affected	-	28 Jul 2015
Huawei Technologies	Affected	-	28 Jul 2015
Kyocera Communications	Affected	-	28 Jul 2015
LG Electronics	Affected	-	28 Jul 2015
Motorola, Inc.	Affected	-	28 Jul 2015
Samsung Mobile	Affected	-	28 Jul 2015
Sony Corporation	Affected	-	28 Jul 2015

Fonte: VULNERABILITY NOTES DATABASE, 2015

A vulnerabilidade “Stagefright” foi atribuída com as seguintes CVEs - *Common Vulnerabilities and Exposures* (CVE DETAILS, 2015):

- CVE-2015-1538
- CVE-2015-1539
- CVE-2015-3824
- CVE-2015-3826
- CVE-2015-3827
- CVE-2015-3828
- CVE-2015-3828
- CVE-2015-3829

Neste cenário único, a Zimperium zLabs não apenas relatou a vulnerabilidade para as equipes do Google, mas também apresentou patches. Considerando a gravidade do problema, o Google agiu prontamente e aplicou os patches para no prazo de 48 horas. Infelizmente, isso é apenas o começo do longo processo de implantação de atualização (VULNERABILITY NOTES DATABASE, 2015). A solução avançada de proteção contra ameaças da Zimperium, “zIPS”, protege seus clientes corporativos da vulnerabilidade “Stagefright” (ZIMPERIUM, 2015).

Além dos clientes da Zimperium, dois grupos de usuários já estão protegidos contra todos os problemas relatados. Usuários do Blackphone, da fabricante SilentCircle, foram protegidos contra essas questões através da atualização do PrivatOS versão 1.1.7 (TECMUNDO, 2014). O navegador Firefox, da Mozilla, que também é afetado, incluiu correções dessa vulnerabilidade desde a versão 38 (LINUX INSIDER, 2015).

Um passo a passo foi criado pela empresa Avast para prevenir a infecção via aplicativos de serviços de mensagens, como o Google Hangouts e o Facebook Messenger. Ele ensina ao usuário como desabilitar a opção para recebimento automático de mensagens nos aplicativos citados (AVAST, 2015).

4.5.2 Stagefright 2.0

Prosseguindo a descoberta de vulnerabilidades na biblioteca “Stagefright” em abril de 2015, Joshua J. Drake continuou pesquisando processamento de mídia em Android. Sua pesquisa constante que se concentrou em ataques remotos contra os dispositivos móveis atuais levou à descoberta de mais uma questão de segurança (DIGITAL TRENDS, 2015).

No Stagefright 2.0, um conjunto de 2 vulnerabilidades que se manifestam no processamento de arquivos MP3 (áudio) e de arquivos MP4 (vídeo) especialmente criados podem que levar à execução de código arbitrário. A primeira vulnerabilidade - em *libutils*, outra biblioteca do Android - gera impactos em praticamente todos os dispositivos Android desde a versão 1.0, lançado em 2008. Foram encontrados métodos para desencadear a primeira vulnerabilidade em dispositivos que executam a versão 5.0 ou superior usando a segunda vulnerabilidade - em *libstagefright*, a biblioteca da pesquisa original (DIGITAL TRENDS, 2015).

O Google atribuiu a vulnerabilidade em *libutils* como: CVE-2015-6602 (CVE DETAILS, 2015). A Zimperium pretende compartilhar informações CVEs para a segunda vulnerabilidade assim que ele estiver disponível (NAKED SECURITY, 2015).

O ataque pode ser acionado à medida que a vulnerabilidade reside no processamento de *metadata*, que são basicamente dados que descrevem dados (TECHTERMS, 2015), dentro dos arquivos MP3 e MP4, de modo que, meramente visualizando a música ou o vídeo provocaria o problema. Uma vez que o principal ataque de MMS foi removido em versões mais recentes do Google Hangouts, um ataque provável seria através do navegador de Internet (DIGITAL TRENDS, 2015).

Um invasor poderia tentar convencer um usuário desavisado a visitar uma URL encurtada que redireciona para um link malicioso controlado, por exemplo, *mobile spear-phishing* ou campanha publicitária mal-intencionada, ou poderia injetar o *exploit* usando técnicas de interceptação de tráfego comum (MITM) para o tráfego de rede sem criptografia destinada ao navegador de Internet. O ataque pode ocorrer até mesmo via apps (*Media Players*, *Instant Messengers*, entre outros) que estão usando a biblioteca vulnerável. (DIGITAL TRENDS, 2015).

A Zimperium notificou à equipe de segurança do Android sobre o assunto no dia 15 de agosto de 2015. Por costume, eles responderam rapidamente e se moveram para remediar. O CVE-2015-6602 foi atribuído para a questão *libutils*. Mas um número CVE para acompanhar a *libstagefright* ainda não foi fornecido. Então, a empresa desenvolveu um patch para atualizar seu aplicativo, o “Stagefright Detector”, criado para detectar essa vulnerabilidade (CVE DETAILS, 2015).

Como cada vez mais pesquisadores têm explorado várias vulnerabilidades que existem dentro da biblioteca “Stagefright” e associadas é esperado a descoberta de mais vulnerabilidades na mesma área. Muitos pesquisadores da comunidade afirmaram que a Google tem respondido aos seus *bugs* relatados, dizendo que eles eram duplicados ou já descobertos internamente. A pesquisa fornecida pela Zimperium nesta área tem sido um catalisador para a mudança (DIGITAL TRENDS, 2015).

Além disso, a Zimperium também enviou uma atualização para a Zimperium Headset Alliance (ZHA), que inclui a maioria dos principais fabricantes de Android. Esta informação pode ser utilizada pelos fabricantes para corrigir dispositivos existentes e futuros (DIGITAL TRENDS, 2015).

4.5.3 Fake ID

Cada aplicativo Android tem a sua própria identidade única, normalmente herdada do desenvolvedor. A equipe de pesquisa da Bluebox Security descobriu uma vulnerabilidade no Android que permite que essas identidades sejam copiadas e usadas para fins maliciosos (ANDROID VULNERABILITIES, 2015).

Chamada de *Fake ID* (Identidade Falsa), a vulnerabilidade permite que aplicativos maliciosos representem aplicativos confiáveis, especialmente reconhecidos sem qualquer notificação do usuário. Isto pode resultar em um amplo espectro de efeitos. Por exemplo, a vulnerabilidade pode ser usada por malwares para escapar do sandbox, que isola informações e a execução do código de um aplicativo (ANDROID DEVELOPERS, 2015), de uma aplicação normal e tomar uma ou mais ações maliciosas, como inserir um “Cavalo de Troia” a um aplicativo através de um disfarce de *plug-in* da Adobe Systems, ou até mesmo ter acesso a dados financeiros e de pagamento através da “personificação” do Google Wallet (TECH TIMES, 2015).

Esta é uma vulnerabilidade generalizada que remonta à liberação do Android 2.1 *Eclair* em janeiro de 2010 e que afeta todos os dispositivos que não são corrigidos para este *bug*, divulgados pelo Google e corrigidos em patches lançados em abril de 2014. Todos os dispositivos com a versão anterior ao Android 4.4 *KitKat* são vulneráveis ao plug-in *WebView* da Adobe Systems, o que permite que um aplicativo malicioso injete um *Trojan* sob a forma deste plug-in *WebView* em outros aplicativos, o que leva a tomar o controle de todo o

aplicativo, manipular todos os dados das apps e ser capaz de fazer qualquer coisa que o app é permitido fazer. Na verdade, o Android 4.4 *KitKat* é vulnerável ao *Fake ID*, mas não especificamente para o plug-in da Adobe, devido a uma mudança no componente *WebView* (TECH TIMES, 2015).

Esta falha funciona através da exploração método do Android de lidar com certificados de identidade, que verifica se um aplicativo é o que parece ser. Certificados de identidade são emitidos através de autoridades de certificado, como a VeriSign. Isso significa que um navegador de Internet confiaria em qualquer certificado emitido pela VeriSign. De acordo com a Bluebox Security (2015), a falha de segurança permite que hackers criem seus próprios certificados de identidade, e em seguida, criem uma reivindicação emitida por uma autoridade de certificação. Depois disso, os hackers podem assinar um aplicativo com o certificado de identidade malicioso e a reivindicação de autoridade de certificação forjada. Outros dispositivos e aplicações que dependem da presença de assinaturas específicas para autenticar uma aplicação podem ser também vulneráveis (TECH TIMES, 2015).

Há um aplicativo publicado na Google Play Store, chamado “FakeID Scanner”, que faz uma varredura no dispositivo à procura por apps que possam ser explorados utilizando esta vulnerabilidade.

4.5.4 Towelroot

A vulnerabilidade *Towelroot* se baseia num método de habilitar o acesso *root* no sistema Android divulgado pelo hacker George Hotz, apelidado como “Geohot”, e causou uma grande polêmica na comunidade Android, já que fornece os privilégios de *root* do usuário em dispositivos que o *root* ainda não tinha sido habilitado. A exploração é feita através da instalação do aplicativo “Towelroot”. George Hotz anunciou sua descoberta no fórum da XDA-Developers[6] e hospedou o arquivo APK do aplicativo no site towelroot.com (GEEKSIDED, 2014).

Como já foi discutido, ao habilitar o acesso em *root* a um dispositivo, o usuário passa a ter o acesso como super-administrador a todos os arquivos e diretórios no dispositivo. A razão de isto não ser o padrão do dispositivo é basicamente para evitar um usuário iniciante de

[6] http://forum.xda-developers.com/showthread.php?t=2783157

inutilizar o seu aparelho acidentalmente ao fazer alterações em arquivos que são cruciais para o seu funcionamento estável (TECHTUDO, 2013).

Um objetivo secundário para dispositivos sendo "não rooteados" por padrão é que isso mantém os aplicativos que não estão construídos no sistema operacional de acessar essas regiões do sistema de arquivos. Esta é uma importante medida de segurança que mantém os aplicativos maliciosos de silenciosamente fazer certas ações sem autorização do usuário, como ligar o GPS e registrar a localização, ligar a câmera e baixar essa informação, ou uma variedade de outras coisas que o pior tipo de malware pode querer fazer (GEEKSIDED, 2014).

Pessoas que gostam de experimentar e possuir um controle maior sobre seu produto preferem ter acesso root para que elas possam personalizar o seu dispositivo mais longe do que é possível sem esse acesso. Elas serão seletivas sobre quais aplicativos eles conceder acesso *root* e os softwares mais usados nestas situações é *Open Source*. Assim, os usuários podem verificar que nada suspeito está acontecendo (GIZMODO, 2013).

Para executar este o processo, o *Towelroot* adquire o acesso *root* explorando uma vulnerabilidade no *Kernel* do Android, que é baseado no Linux e é um componente do dispositivo que, basicamente, age como um intermediário para fazer com que o sistema operacional se comunique com o hardware (THREADPOST, 2014).

O *Kernel* traduz os pedidos do software em comandos para o processador e outros componentes internos e vice-versa. Cada dispositivo terá um *Kernel* único uma vez que cada dispositivo está usando um equipamento diferente, mas existem componentes chave do *Kernel* que não são susceptíveis de serem alterados, desde que o sistema operacional seja o Android (GEEKSIDED, 2014).

Por esta razão, muitas pessoas reagiram à liberação do aplicativo “Towelroot” com grande entusiasmo, porque quase todos os dispositivos Android compartilham esta façanha. Esse aplicativo pode “rootear” quase todos os dispositivos Android. Pelo fato da simplicidade do processo ser executado por meio de um aplicativo, o *root* é realmente fácil de fazer. Normalmente o root é obtido através da introdução de comandos e upload de arquivos para o dispositivo a partir de um computador, sendo que este é um processo muito mais complicado (ESCOLA ANDROID, 2014).

O criador desta ferramenta, George Hotz, é um hacker *White Hat* muito conhecido pelas suas façanhas anteriores, como o desbloqueio de iPhones e do PlayStation 3. Por esta razão, o aplicativo “Towelroot” é, ao que tudo indica, confiável e extremamente bom no que ele tem a intenção de fazer, que é acesso *root* para entusiastas e desenvolvedores Android moderadamente competentes ou mais experientes (CANALTECH, 2014). Porém a ferramenta carrega uma vunerabilidade consigo. Esta vulnerabilidade de segurança CVE-2014-3153 (CVE DETAILS, 2014), quando explorada, pode permitir que qualquer aplicativo consiga o acesso *root* e, com isto, privilégios de um super-administrador, permitindo um invasor ignorar o módulo de segurança do Android e executar códigos maliciosos, como recuperar vários arquivos e informações sensíveis, acessar aplicações de proteção de dados da empresa e inserir um persistente *backdoor* no dispositivo para ser usado mais tarde para outras atividades de ataque (THREADPOST, 2014).

> O aplicativo executa algum código, o código trava o sistema Android e consegue deixá-lo confuso, e no seu estado confuso o sistema pensa que o aplicativo deve possuir acesso *root*, em seguida, o aplicativo instala algo para permitir que outros aplicativos assumam também os privilégios de *root* (BITMASTRO, 2014, tradução nossa).

Isso tudo acontece sem o usuário saber uma coisa. Quando se consegue o acesso de propósito, o software informa toda vez que um aplicativo quer privilégios de *root* e permite ao usuário autorizar, negar ou dar um período de estágio de acesso ao aplicativo fornecido. Se esta façanha é alcançada por alguém menos confiável, não haverá quaisquer indicações óbvias de que isso aconteceu (GEEKSIDED, 2014).

Infelizmente, isto seria muito fácil de fazer. Se um desenvolvedor de software mal intencionado tem um app, que muitas pessoas o baixaram, descobrir esse *exploit*, ele poderia criar uma atualização para o seu aplicativo que executa um script, como descrito anteriormente, causando muitos problemas para o usuário. A parte do *Kernel* que George Hotz aproveitou é tão básica que está presente em quase todos os *Kernel* Linux de hoje. Não apenas aqueles que aparecem em dispositivos Android, mas em várias distribuições Linux que são executadas em computadores e servidores web. A façanha foi descoberta por um revisor de código adolescente anônimo conhecido como "Pinkie Pie" e foi divulgada para encorajar os desenvolvedores Linux para atualizar o *Kernel* (TOM’S GUIDE, 2014).

Este problema foi corrigido e não demorou muito para o os usuários de PC e servidores para receber o patch, porém não é assim tão fácil para dispositivos Android, onde as atualizações

das versões do sistema operacional têm aqueles problemas citados anteriormente. Demasiadas vezes, os dispositivos mais antigos são abandonados por seus fabricantes, logo, é muito improvável que surgirá uma atualização para corrigir esta vulnerabilidade. Até mesmo para dispositivos mais recentes, a obtenção destas atualizações é demorada (SALMI, 2014).

Para se proteger contra isso, os usuários só devem instalar aplicativos Android do Google Play Store e certificar-se de seus dispositivos não pode aceitar software de “fontes desconhecidas”. Um bom aplicativo de segurança Android também pode ser capaz de detectar esse código de exploração em software baixado (SECURITY WEEK, 2014).

5 CONCLUSÃO

O tema desta pesquisa é segurança. O problema que gerou o trabalho realizado foi: Quais as vulnerabilidades mais comuns que foram reportadas no sistema Android e os seus mecanismos para garantir a segurança?

A partir deste problema objetivou-se estudar as vulnerabilidades principais relacionadas do sistema operacional Android entre a API 18 e a API 22 (exceto API 20).

Justifica-se este estudo frente a liderança que o Android exerce dentro do mercado e a facilidade de pesquisa sobre o mesmo por ele ser um sistema de código aberto (*Open Source*), o que possibilitou o cumprimento dos principais objetivos adotados ao início do projeto.

Os principais resultados apontados pela pesquisa foram relacionados as dificuldades enfrentadas pelos desenvolvedores ao lidar com a proteção dos mesmos, devido aos aplicativos mal-intencionados ou falhas no próprio SO. Ainda foi apontada necessidade e importância de melhorar o processo de comunicação entre as multinacionais fabricantes de aparelhos que utilizam o Android junto ao Google para sanar estes problemas.

Entende-se que a hipótese inicial foi confirmada e que a pesquisa atendeu ao seu objetivo principal. Apesar de apontar algumas vulnerabilidades, o sistema Android pode ser considerado plataforma eficiente e sua interface atrai usuários técnicos e leigos e este mercado crescente gera investimentos em ações de melhoria da segurança da informação.

Esta pesquisa se mostrou como uma grande oportunidade para aumentar os conhecimentos e colocar em prática o que foi aprendido ao longo dos anos de estudo durante o curso de Ciência da Computação realizado na Universidade FUMEC.

Apesar de não ser foco a generalização dos resultados aqui encontrados, entende-se que o estudo contribui na discussão do tema segurança em especial quanto ao sistema Android e ainda gera novos questionamentos. Sugere-se, como estudo futuro analisar a usabilidade nos diferentes dispositivos Android, de forma comparativa, e o que isto pode influenciar na segurança do sistema.

REFERÊNCIAS

ANDROID. **Android**. 2015. Disponível em: <https://www.android.com>. Acesso em: 18 set. 2015.

ANDROID. **Android - History**. 2015. Disponível em: <https://www.android.com/history/>. Acesso em: 18 set. 2015.

ANDROID. **Android - 4.3 Jelly Bean**. 2015. Disponível em: <https://www.android.com/versions/jelly-bean-4-3/>. Acesso em: 18 set. 2015.

ANDROID. **Android - 4.4 KitKat**. 2015. Disponível em: <https://www.android.com/versions/kit-kat-4-4/>. Acesso em: 18 set. 2015.

ANDROID. **Android - 5.0 Lollipop**. 2015. Disponível em: <https://www.android.com/versions/lollipop-5-0/>. Acesso em: 18 set. 2015.

ANDROID. **Android - 6.0 Marshmallow**. 2015. Disponível em: <https://www.android.com/versions/marshmallow-6-0/>. Acesso em: 18 set. 2015.

ANDROID DEVELOPERS. **Android 6.0 Marshmallow**. 2015. Disponível em: <http://developer.android.com/intl/pt-br/about/versions/marshmallow/index.html>. Acesso em: 01 nov. 2015.

ANDROID DEVELOPERS. **Android Developers - Homepage**. 2015. Disponível em: <https://developer.android.com/>. Acesso em: 18 set. 2015.

ANDROID DEVELOPERS. **Android KitKat**. 2013. Disponível em: <http://developer.android.com/intl/pt-br/about/versions/kitkat.html>. Acesso em: 01 nov. 2015.

ANDROID DEVELOPERS. **Dashboards**. 2015. Disponível em: <https://developer.android.com/about/dashboards/index.html>. Acesso em: 18 set. 2015.

ANDROID DEVELOPERS. **Full Screen**. 2013. Disponível em: <http://developer.android.com/intl/pt-br/design/patterns/fullscreen.html>. Acesso em: 18 set. 2015.

ANDROID DEVELOPERS. **Jelly Bean**. 2013. Disponível em: <http://developer.android.com/intl/pt-br/about/versions/jelly-bean.html>. Acesso em: 01 nov. 2015.

ANDROID DEVELOPERS. **Material Design**. 2014. Disponível em: <http://developer.android.com/intl/pt-br/design/material/index.html>. Acesso em: 18 set. 2015.

ANDROID DEVELOPERS. **Security Tips**. 2015. Disponível em: <http://developer.android.com/intl/pt-br/training/articles/security-tips.html>. Acesso em: 18 set. 2015.

ANDROID OPEN SOURCE PROJECT. **Android Open Source Project (AOSP) - Homepage**. 2015. Disponível em: <https://source.android.com/>. Acesso em: 18 set. 2015.

ANDROID OPEN SOURCE PROJECT. **Media - Android Open Source Project (AOSP)**. 2015. Disponível em: <https://source.android.com/devices/media/index.html>. Acesso em: 18 set. 2015.

ANDROID VULNERABILITIES (Cambridge, ING). University of Cambridge. **Android Vulnerabilities**. 2015. Disponível em: <http://www.androidvulnerabilities.org/>. Acesso em: 23 out. 2015.

ANDROIDBEAT. **Android 4.4 KitKat is official; brings new launcher, lower RAM consumption, improved touch feedback and more**. 2013. Disponível em: <http://www.androidbeat.com/2013/10/android-4-4-kitkat-official/>. Acesso em: 13 nov. 2015.

ANDROIDPIT. **Android 5.1: funções e mudanças da nova versão do OS**. 2015. Disponível em: <http://www.androidpit.com.br/android-5-1-lancamento-atualizacao-funcoes>. Acesso em: 01 nov. 2015.

ANDROIDPIT. **Android 6.0 Marshmallow - review e todas as funções**. 2015. Disponível em: <http://www.androidpit.com.br/android-m-dispositivos-data-lancamento-funcoes>. Acesso em: 13 nov. 2015.

ANDROIDPIT. **Design do Android 4.4 KitKat: visão geral das mudanças**. 2013. Disponível em: <http://www.androidpit.com.br/design-android-4-4-kitkat-visao-geral-mudancas>. Acesso em: 01 nov. 2015.

ARS TECHNICA. **950 million Android phones can be hijacked by malicious text messages**. 2015. Disponível em: <http://arstechnica.com/security/2015/07/950-million-android-phones-can-be-hijacked-by-malicious-text-messages/>. Acesso em: 01 nov. 2015.

AVAST. **Mobile Security: Stagefright - nova vulnerabilidade Android coloca o seu dispositivo em risco**. 2015. Disponível em: <https://www.avast.com/pt-br/faq.php?article=AVKB230>. Acesso em: 01 nov. 2015.

BEAL, Vangie. **Adware**. Disponível em: <http://www.webopedia.com/TERM/A/adware.html>. Acesso em: 18 set. 2015.

BLACK HAT USA 2015. **Black Hat USA 2015 - Briefings**. 2015. Disponível em: <https://www.blackhat.com/us-15/schedule/briefings-5.html>. Acesso em: 01 nov. 2015.

BLUEBOX SECURITY. **Bluebox Security - Homepage**. 2015. Disponível em: <https://bluebox.com/>. Acesso em: 01 nov. 2015.

BITMASTRO. **Samsung S5 root is live!** 2014. Disponível em: <https://www.reddit.com/r/Android/comments/287n3t/samsung_s5_root_is_live/ci8b17v>. Acesso em: 01 nov. 2015.

BORDIN, Maycon Viana. **Introdução a Arquitetura Android**. 2012. 1 v. TCC (Graduação) - Curso de Sistemas de Informação, Sociedade Educacional Três de Maio, Três de Maio - RS, 2012. Disponível em: <http://sites.setrem.com.br/stin/2012/anais/Maycon.pdf>. Acesso em: 13 nov. 2015.

BORGES, Ciro Fernando Preto; BENTO, Paulo Diego Nogueira. **Segurança de Redes Utilizando Honeypot.** 2006. 63 f. Monografia (Especialização) - Curso de Engenharia da Computação, Instituto de Estudos Superiores da Amazônia, Belém, 2006. Disponível em: <http://www3.iesam-pa.edu.br/ojs/index.php/computacao/article/viewFile/83/78>. Acesso em: 13 nov. 2015.

CANALTECH. **Geohot, o hacker que destravou o PlayStation 3, é contratado pelo Google**. 2014. Disponível em: <http://canaltech.com.br/noticia/google/Geohot-o-hacker-que-destravou-o-PlayStation-3-e-contratado-pelo-Google/>. Acesso em: 01 nov. 2015.

CANALTECH. **Google lança Android 5.1 e atualização começa a ser distribuída ainda hoje**. 2015. Disponível em: <http://canaltech.com.br/noticia/android/Google-lanca-Android-51-e-atualizacao-comeca-a-ser-distribuida-ainda-hoje/>. Acesso em: 13 nov. 2015.

CANALTECH. **Google não fará mais atualizações de segurança para Android 4.3 e anteriores.** 2015. Disponível em: <http://canaltech.com.br/noticia/android/Google-nao-fara-mais-atualizacoes-de-seguranca-para-Android-43-e-anteriores/>. Acesso em: 13 nov. 2015.

CÁRDENAS, Daniel Stevens Torres. **Conceitos Básicos e Práticos do Android**. 2011. Disponível em: <http://www.docdroid.net/roWXh2g/apostila-pilula-android.pdf.html>. Acesso em: 13 nov. 2015.

CAREY NACHENBERG (Brasil). **Symantec Corporation. Uma janela para a segurança nos dispositivos móveis**. 2011. Disponível em: <https://www.symantec.com/content/pt/br/enterprise/images/theme/mobiletrends/mobile_device_security-pt.pdf>. Acesso em: 14 ago. 2015.

CISCO SYSTEMS INC. **Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update 2014-2019 White Paper - Cisco**. 2013. Disponível em: <http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.html>. Acesso em: 18 set. 2015.

CNET. **How to enable dial-pad autocomplete in Android 4.3**. 2013. Disponível em: <http://www.cnet.com/how-to/how-to-enable-dial-pad-autocomplete-in-android-4-3/>. Acesso em: 13 nov. 2015.

CVE DETAILS. **Google Android: CVE security vulnerabilities, versions and detailed reports**. 2015. Disponível em: <http://www.cvedetails.com/product/19997/Google-Android.html?vendor_id=1224>. Acesso em: 23 out. 2015.

CVE DETAILS. **Google Android: List of security vulnerabilities**. 2015. Disponível em: <http://www.cvedetails.com/vulnerability-list/vendor_id-1224/product_id-19997/Google-Android.html>. Acesso em: 23 out. 2015.

CVE DETAILS. **Vulnerability Details: CVE-2014-3153**. 2014. Disponível em: <http://www.cvedetails.com/cve/2014-3153>. Acesso em: 23 out. 2015.

DEF CON 23. **DEF CON 23 Hacking Conference - Shedule**. 2015. Disponível em: <https://www.defcon.org/html/defcon-23/dc-23-schedule.html>. Acesso em: 01 nov. 2015.

DIGITAL TRENDS. **Everything you need to know about Stagefright 2.0, Android's newest security threat**. 2015. Disponível em: <http://www.digitaltrends.com/mobile/stagefright-2-0-hack-news/>. Acesso em: 01 nov. 2015.

ESCOLA ANDROID. **TowelRoot Libera Acesso Root em "Todos" os Dispositivos**. 2014. Disponível em: <http://www.escolaandroid.com/towelroot-acesso-root-em-todos-dispositivos/>. Acesso em: 01 nov. 2015.

FERREIRA, Ricardo; SUPERDOWNLOADS. **Outras maneiras além do Google Play de instalar aplicativos no Android**. 2013. Disponível em: <http://www.superdownloads.com.br/materias/6850-outras-maneiras-alem-do-google-play-de-instalar-aplicativos-no-android.htm>. Acesso em: 01 nov. 2015.

FORBES. **Stagefright: It Only Takes One Text To Hack 950 Million Android Phones**. 2015. Disponível em: <http://www.forbes.com/sites/thomasbrewster/2015/07/27/android-text-attacks/>. Acesso em: 01 nov. 2015.

GASPAR, Philipe. **Pragas eletrônicas: ainda não estamos livres delas**. 2007. Disponível em: <http://www.philipegaspar.com/2007/07/pragas-eletronicas-ainda-nao-estamos_22.html>. Acesso em: 13 nov. 2015.

GEEKSIDED. **"Towelroot" exploit reveals security nightmare for Android**. 2014. Disponível em: <http://geeksided.com/2014/06/16/towelroot-exploit-reveals-security-nightmare-android/>. Acesso em: 01 nov. 2015.

GIZMODO. **9 Reasons to Root Your Android Device**. 2013. Disponível em <http://gizmodo.com/5982287/reasons-to-root-your-android-device>. Acesso em: 13 nov. 2015.

GIZMODO BRASIL. **10 motivos para fazer root no seu Android**. 2012. Disponível em: <http://gizmodo.uol.com.br/10-motivos-para-fazer-root-no-seu-android/>. Acesso em: 13 nov. 2015.

GOMES, Rafael Caveari; FERNANDES, Jean Alves R.; FERREIRA, Vinicius Corrêa. **Sistema Operacional Android**. 2012. 32 f. TCC (Graduação) - Curso de Engenharia de Telecomunicações, Universidade Federal Fluminense, Niterói - RJ, 2012. Disponível em: <http://www.midiacom.uff.br/~natalia/2012-1-sisop/tgrupo1.pdf>. Acesso em: 13 nov. 2015.

GRAVEHEART, Paulo. **CyanogenMod ultrapassa 1 milhão de instalações**. 2012. Disponível em: <https://tecnoblog.net/88366/cyanogen-1-milhao-instalacoes/>. Acesso em: 01 nov. 2015.

GRIFFIN, B. **An Introduction to Viruses and Malicious Code, Part One: Overview**. 2000. Disponível em: <http://www.securityfocus.com/infocus/1188>. Acesso em: 13 nov. 2015.

GSMA INTELLIGENCE. **Definitive data and analysis for the mobile industry**. 2015. Disponível em: <https://gsmaintelligence.com/>. Acesso em: 18 set. 2015.

GUIMARÃES, Gleyser. **A história do sistema operacional Android**. 2013. Disponível em: <http://www.dsc.ufcg.edu.br/~pet/jornal/agosto2013/materias/historia_da_computacao.html>. Acesso em: 18 set. 2015.

INÁCIO, Leandro Rodrigues; GOMES, Antônio Ricardo Leocádio. **Uma abordagem sobre problemas de Segurança da Informação por meio do desenvolvimento de aplicações maliciosas para Android**. E-xacta, Belo Horizonte, v. 7, n. 2, p.87-106, 30 nov. 2014. Disponível em: <http://revistas.unibh.br/index.php/dcet/article/viewFile/1344/737>. Acesso em: 14 ago. 2015.

INFO WESTER. **Vírus de computador e outros malwares: o que são e como agem**. 2013. Disponível em: <http://www.infowester.com/malwares.php>. Acesso em: 18 set. 2015.

ISHIMI, Valgney Cherri; UCHÔA, Joaquim Quinteiro. **Pragas virtuais em Linux.** 2005. Disponível em: <http://repositorio.ufla.br/bitstream/1/9648/1/ARTIGO_Pragas_virtuais_em_linux.pdf>. Acesso em: 13 nov. 2015.

KASPERSKY LAB. **The very first mobile malware: how Kaspersky Lab discovered Cabir**. 2014. Disponível em: <http://www.kaspersky.com/about/news/virus/2014/The-very-first-mobile-malware-how-Kaspersky-Lab-discovered-Cabir>. Acesso em: 18 set. 2015.

LECHETA, Ricardo R. **Google Android: Aprenda a criar aplicações para dispositivos móveis com o Android SDK**. 4. ed. São Paulo: Novatec, 2015. 1016 p.

LIFEHACKER. **Everything You Need to Know About Rooting Your Android Phone**. 2013. Disponível em: <http://lifehacker.com/5789397/the-always-up-to-date-guide-to-rooting-any-android-phone>. Acesso em: 13 nov. 2015.

LINUX INSIDER. **Super-Scary Android Flaw Found**. 2015. Disponível em: <http://www.linuxinsider.com/story/82315.html>. Acesso em: 01 nov. 2015.

LUDWIG, Adrian. **Adrian Ludwig - about the closure of the Jelly Bean updates - Google+ Profile**. 2015. Disponível em: <https://plus.google.com/+AdrianLudwig/posts/1md7ruEwBLF>. Acesso em: 01 nov. 2015.

MACK, Roger Schneider. **Sistema de recomendação baseado na Localização e Perfil utilizando a plataforma Android**. 2010. 56 f. TCC (Graduação) - Curso de Ciência da Computação, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2010. Disponível em: <https://www.lume.ufrgs.br/bitstream/handle/10183/28328/000767836.pdf>. Acesso em: 13 nov. 2015.

MANN, Steve. **Definition of "Wearable Computer"**. 1998. Disponível em: <http://wearcomp.org/wearcompdef.html>. Acesso em: 02 out. 2015.

MARTINS, Ricardo. **Hacker.** 2015. Disponível em: <http://knoow.net/ciencinformtelec/informatica/hacker/>. Acesso em: 13 nov. 2015.

MEIO BIT. **Google não corrigirá falha de segurança que afeta mais de um bilhão de Androids**. 2015. Disponível em: <http://meiobit.com/307066/google-correcao-falha-seguranca-android-um-bilhao-aparelhos-afetados-nao-sera-corrigida/>. Acesso em: 13 nov. 2015.

MEYER, Maximiliano. **A História do Android**. 2015. Disponível em: <https://www.oficinadanet.com.br/post/13939-a-historia-do-android>. Acesso em: 18 set. 2015.

MICROSOFT. **Spyware: perguntas frequentes**. 2015. Disponível em: <http://windows.microsoft.com/pt-br/windows/spyware-faq>. Acesso em: 13 nov. 2015.

MOBILEXPERT. **Build Developer Preview do Android 6.0 Marshmallow exibe nova animação de inicialização.** 2015. Disponível em: <http://mobilexpert.com.br/apps/outros/materias/12968/build-developer-preview-do-android-60-marshmallow-exibe-nova-animacao-de-inicializacao>. Acesso em: 13 nov. 2015.

NAKED SECURITY. **Google emite correções para Android Stagefright 2 (para alguns usuários)**. 2015. Disponível em: <https://nakedsecurity.sophos.com/pt/2015/10/06/google-issues-android-patches-for-stagefright-2-for-some-users/>. Acesso em: 01 nov. 2015.

NET MARKET SHARE. **Market share for mobile, browsers, operating systems and search engines**. 2015. Disponível em: <https://netmarketshare.com/>. Acesso em: 18 set. 2015.

NORTON SYMANTEC. **Spear Phishing: um golpe, não um esporte**. 2011. Disponível em: <http://br.norton.com/spear-phishing-scam-not-sport/article>. Acesso em: 01 nov. 2015.

OFICINA DA NET. **Root no Android vale a pena?** 2013. Disponível em: <https://www.oficinadanet.com.br/post/11175-root-no-android-vale-a-pena>. Acesso em: 13 nov. 2015.

OLHAR DIGITAL. **Entenda o que muda no Android 4.3**. 2013. Disponível em: <http://olhardigital.uol.com.br/noticia/entenda-o-que-muda-no-android-4-3/36118>. Acesso em: 18 out. 2015.

OLHAR DIGITAL. **Root para Android: saiba o que é e como fazer**. 2014. Disponível em: <http://olhardigital.uol.com.br/noticia/root-para-android-saiba-o-que-e-e-como-fazer/40421>. Acesso em: 18 out. 2015.

OPEN HANDSET ALLIANCE. **Allience Members**. 2015. Disponível em: <http://www.openhandsetalliance.com/oha_members.html>. Acesso em: 18 set. 2015.

OPEN HANDSET ALLIANCE. **Open Handset Alliance - Homepage**. 2015. Disponível em: <http://www.openhandsetalliance.com/>. Acesso em: 18 set. 2015.

PEREIRA, Lúcio Camilo Oliva; SILVA, Michel Lourenço da. **Android para Desenvolvedores**. 2. ed. Rio de Janeiro: Brasport, 2012. 248 p. CD-ROM.

PEREIRA JÚNIOR, Marcos Ronaldo. **Estudo da correlação entre métricas de código fonte do sistema Android e seus aplicativos**. 2014. 48 f. TCC (Graduação) - Curso de Engenharia de Software, Universidade de Brasília, Brasília, 2014. Disponível em: <https://fga.unb.br/articles/0001/0314/tcc2_marcos.pdf>. Acesso em: 13 nov. 2015.

RAYMOND, Eric S. **The New Hacker's Dictionary**. 3. ed. Cambridge, EUA: The MIT Press, 1996. 547 p.

ROCHA, Eduardo. **Saiba tudo sobre ROOT no Android**. 2013. Disponível em: <http://dudurochatec.com.br/saiba-tudo-sobre-root/>. Acesso em: 01 nov. 2015.

ROTONDO, Gustavo. **Segurança e integridade de dados em sistemas operacionais móveis:** Um estudo de caso sobre a plataforma Android. 2014. 10 f. Monografia (Especialização) - Curso de Engenharia da Computação, UNIMPAMPA, Bagé, 2014. Disponível em: <http://periodicos.unesc.net/index.php/sulcomp/article/view/1774/1679>. Acesso em: 14 ago. 2015.

RUSSO, Rafael. **Hackers Black Hats, White Hats e Gray Hats, qual a diferença?** 2013. Disponível em: <http://escreveassim.com.br/2013/04/22/hackers-black-white-grey-hat/>. Acesso em: 13 nov. 2015.

SALMI, Deborah; AVAST. **Samsung Galaxy S5 and other popular phones vulnerable to "TowelRoot" Android exploit**. 2014. Disponível em: <https://blog.avast.com/2014/06/20/samsung-galaxy-s5-and-other-popular-phones-vulnerable-to-towelroot-android-exploit/>. Acesso em: 01 nov. 2015.

SANTOS, Jonas Matias. **Hackers Mocinhos ou Bandidos? Uma Análise Dentro da Hierarquia**. 2010. Disponível em: <http://www.olharcientifico.kinghost.net/index.php/olhar/article/viewFile/50/35>. Acesso em: 13 nov. 2015.

SCHWARTZ, Mathew J. **Android Hackers Craft GingerMaster Rootkit**. 2011. Disponível em: <http://www.informationweek.com/mobile/android-hackers-craft-gingermaster-rootkit/d/d-id/1099766>. Acesso em: 18 set. 2015.

SCOTA, Daniel Fernando; ANDRADE, Gil Eduardo de; XAVIER, Rafael da Costa. **Configuração de Rede sem Fio e Segurança no Sistema Operacional Android**. 2010. 23 f. Monografia (Especialização) - Curso de Redes e Segurança de Sistemas, PUC-PR, Curitiba, 2010. Disponível em: <http://www.ppgia.pucpr.br/~jamhour/RSS/TCCRSS08B/Daniel%20Fernando%20Scota%20-%20Artigo.pdf>. Acesso em: 14 ago. 2015.

SECURITY WEEK. **TowelRoot Vulnerability Could Lead to Attacks on Android Devices: Researcher**. 2014. Disponível em: <http://www.securityweek.com/towelroot-vulnerability-could-lead-attacks-android-devices-researcher>. Acesso em: 01 nov. 2015.

SERALO; SOCIAL COMPARE. **Android versions comparison**. 2015. Disponível em: <http://www.socialcompare.com/en/comparison/android-versions-comparison>. Acesso em: 14 ago. 2015.

SILICON ANGLE. **The elephant in the room: Study confirms Android devices vulnerable due to lack of patches**. 2015. Disponível em: <http://siliconangle.com/blog/2015/10/14/the-elephant-in-the-room-study-confirms-android-devices-vulnerable-due-to-lack-of-patches/>. Acesso em: 01 nov. 2015.

TALKDROID. **Veja neste artigo as novas funções do Android 5.1 Lollipop**. 2015. Disponível em: <http://www.talkdroid.com.br/?p=915>. Acesso em: 13 nov. 2015.

TECH TIMES. **Android Fake ID bug allows hackers to take over people's phones**. 2015. Disponível em: <http://www.techtimes.com/articles/11602/20140801/android-fake-id-bug-allows-hackers-to-take-over-people-s-phones.htm>. Acesso em: 01 nov. 2015.

TECHTERMS. **Malware**. Disponível em: <http://techterms.com/definition/malware>. Acesso em: 18 set. 2015.

TECHTERMS. **Metadata**. Disponível em: <http://techterms.com/definition/metadata>. Acesso em: 18 set. 2015.

TECHTUDO. **Android 4.3: entenda as novidades do novo sistema do Google**. 2015. Disponível em: <http://www.techtudo.com.br/noticias/noticia/2013/07/android-43-entenda-as-novidades-do-novo-sistema-do-google.html>. Acesso em: 13 nov. 2015.

TECHTUDO. **Confira o Android 4.4, a evolução mais completa e rápida do sistema**. 2015. Disponível em: <http://www.techtudo.com.br/tudo-sobre/android-4-4.html>. Acesso em: 13 nov. 2015.

TECHTUDO. **Conheça os riscos de se fazer root em um aparelho Android**. 2013. Disponível em: <http://www.techtudo.com.br/artigos/noticia/2013/04/conheca-os-riscos-de-se-fazer-root-em-um-aparelho-android.html>. Acesso em: 13 nov. 2015.

TECHTUDO. **I/O 2015: Google lança Preview do Android M, Android Pay, Fotos e mais**. 2015. Disponível em: <http://www.techtudo.com.br/noticias/noticia/2015/05/google-io-2015-android-m.html>. Acesso em: 13 nov. 2015.

TECHTUDO. **Seis dicas para identificar se um aplicativo do Google Play é seguro ou não**. 2015. Disponível em: <http://www.techtudo.com.br/dicas-e-tutoriais/noticia/2013/04/seis-dicas-para-identificar-se-um-aplicativo-do-google-play-e-seguro-ou-nao.html>. Acesso em: 01 nov. 2015.

TECMUNDO. **Análise: Android 6.0 Marshmallow para smartphones**. 2015. Disponível em: <http://www.tecmundo.com.br/android-marshmallow/88790-analise-android-6-0-marshmallow-smartphones.htm>. Acesso em: 01 nov. 2015.

TECMUNDO. **Android reúne 97% dos malwares, mas só 0,1% está na Google Play**. 2014. Disponível em: <http://www.tecmundo.com.br/malware/52026-android-reune-97-dos-malwares-mas-so-0-1-esta-na-google-play.htm>. Acesso em: 01 nov. 2015.

TECMUNDO. **Blackphone, o “celular blindado”, começa a ser entregue a seus compradores**. 2014. Disponível em: <http://www.tecmundo.com.br/seguranca/58420-blackphone-celular-blindado-comeca-entregue-compradores.htm>. Acesso em: 01 nov. 2015.

TECMUNDO. **Google diz que ameaça de vírus a celulares é desprezível**. 2014. Disponível em: <http://www.tecmundo.com.br/google/54198-google-diz-que-ameaca-de-virus-a-celulares-e-desprezivel.htm>. Acesso em: 01 nov. 2015.

TECMUNDO. **Google explica por que "abandonou" as versões mais antigas do Android**. 2015. Disponível em: <http://www.tecmundo.com.br/android/73116-google-explica-abandonou-versoes-antigas-android.htm>. Acesso em: 01 nov. 2015.

TECMUNDO. **Google toma decisão que vai prejudicar milhões de usuários do Android**. 2015. Disponível em: <http://www.tecmundo.com.br/android/72476-google-toma-decisao-prejudicar-milhoes-usuarios-android.htm>. Acesso em: 01 nov. 2015.

TECMUNDO. **Linha do Tempo: por dentro da evolução do Android**. 2015. Disponível em: <http://www.tecmundo.com.br/android/82344-linha-tempo-dentro-evolucao-do-sistema-android.htm>. Acesso em: 01 nov. 2015.

TECMUNDO. **Pior do que pensávamos: Google Play tem mais de 30 mil apps maliciosos**. 2015. Disponível em: <http://www.tecmundo.com.br/android/85697-pior-pensavamos-google-play-tem-30-mil-apps-maliciosos.htm>. Acesso em: 01 nov. 2015.

TECNOBLOG. **Google anuncia Android 4.3 Jelly Bean**. 2013. Disponível em: <https://tecnoblog.net/130951/google-android-4-3/>. Acesso em: 13 nov. 2015.

TECNOBLOG. **O que há de novo (até agora) no Android 6.0 Marshmallow**. 2015. Disponível em: <https://tecnoblog.net/130951/google-android-4-3/>. Acesso em: 13 nov. 2015.

TECNOLOGIA E TAL. **Android Lollipop: Conheça os Novos Recursos**. 2014. Disponível em: <http://www.tecnologiaetal.com.br/2014/11/android-lollipop-conheca-os-novos-recursos/>. Acesso em: 13 nov. 2015.

THREATPOST. **Android Root Access Vulnerability Affecting Most Devices**. 2014. Disponível em: <https://threatpost.com/android-root-access-vulnerability-affecting-most-devices/106683/>. Acesso em: 13 nov. 2015.

TOM’S GUIDE. **Nearly All Android Devices Vulnerable to Rooting Attack**. 2014. Disponível em: <http://www.tomsguide.com/us/android-towelroot-linux-flaw,news-19012.html>. Acesso em: 13 nov. 2015.

TREND MICRO. **Masque, FakeID, and Other Notable Mobile Threats of 2H 2014**. 2014. Disponível em: <http://www.trendmicro.com/vinfo/us/security/news/mobile-safety/masque-fakeid-and-other-notable-mobile-threats-of-2h-2014>. Acesso em: 18 set. 2015.

ULBRICH, Henrique Cesar; DELLA VALLE, James. **Universidade Hacker**. 4. ed. São Paulo: Digerati Books, 2004. 348 p.

UOL. **O que é malware, adware, cavalo de Troia e spyware**. 2013. Disponível em: <http://seguranca.uol.com.br/antivirus/dicas/curiosidades/o-que-e-malware-adware-cavalo-troia-spyware.html>. Acesso em: 18 set. 2015.

VERISIGN. **VeriSign - Homepage**. 2015. Disponível em: <http://www.verisign.com/pt_BR/>. Acesso em: 01 nov. 2015.

VIDAS, Timothy; VOTIPKA, Daniel; CHRISTIN, Nicolas. **All Your Droid Are Belong To Us: A Survey of Current Android Attacks.** 2011. Disponível em: <https://www.usenix.org/legacy/event/woot11/tech/final_files/Vidas.pdf>. Acesso em: 01 nov. 2015.

VULNERABILITY NOTES DATABASE. **Vulnerability Note VU#924951:** Android Stagefright contains multiple vulnerabilities. 2015. Disponível em: <http://www.kb.cert.org/vuls/id/924951>. Acesso em: 01 nov. 2015.

WINDOWSTEAM. **A Google não garante mais a segurança de usuários da versão 4.3 do Android e anteriores**. 2015. Disponível em: <https://www.windowsteam.com.br/google-nao-garante-mais-seguranca-de-usuarios-da-versao-4-3-android-e-anteriores/>. Acesso em: 13 nov. 2015.

XAVIER, Andressa. **O que é Spyware?** 2008. Disponível em: <http://www.tecmundo.com.br/spyware/29-o-que-e-spyware-.htm>. Acesso em: 13 nov. 2015.

ZIMPERIUM. **zIPS - Advanced Mobile Threat Defense**. 2015. Disponível em: <https://www.zimperium.com/zips-mobile-ips>. Acesso em: 01 nov. 2015.