UNIVERSIDADE FEEVALE

ROBSON DE PAIVA

UM ESTUDO SOBRE PROGRAMAS DE BUG BOUNTY E LEVANTAMENTO DE VULNERABILIDADES PARA O AUXÍLIO DE PENTESTS

Novo Hamburgo

2020

ROBSON DE PAIVA

UM ESTUDO SOBRE PROGRAMAS DE BUG BOUNTY E LEVANTAMENTO DE VULNERABILIDADES PARA O AUXÍLIO DE PENTESTS

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do grau de Bacharel em Ciência da Computação pela Universidade Feevale

Orientador: Me. Vandersilvio Da Silva

Novo Hamburgo

2020

AGRADECIMENTOS

Gostaria de agradecer a todos que de alguma forma contribuíram para a realização desse trabalho de conclusão, em especial:

A Deus, por ter me dado saúde e força para superar as dificuldades.

Ao Prof. Me. Vandersilvio da Silva, pela oportunidade e apoio na elaboração deste trabalho.

Ao Prof. Dr. Daniel Dalalana Bertoglio pelo apoio na pesquisa do trabalho.

E aos demais professores, que contribuíram em todos esses anos acadêmicos no meu crescimento pessoal e profissional.

Aos meus pais, pelo amor, incentivo e apoio incondicional.

A minha esposa e meu filho, por toda paciência, compreensão, carinho e amor, e por me ajudar muitas vezes a achar soluções quando elas pareciam não aparecer.

RESUMO

O uso do BBP (*Bug Bounty Program*) nas empresas de *software* tem se tornado cada vez mais frequente, visto que muitas empresas adaptaram significativamente a sua abordagem, integrando mais abertamente informações de vulnerabilidade obtidas externamente. Empresas como *Microsoft* e *Facebook* desenvolveram programas estruturados, no qual *bug hunters* enviam as vulnerabilidades encontradas e em troca recebem uma recompensa já predefinida, dependendo do *bug* o preço da recompensa pode variar de 100 até 100.000 dólares. Sendo assim, este trabalho tem como objetivo realizar um levantamento e unificar as informações sobre determinadas plataformas de *Bug Bounty Program*, a fim de fornecer informações melhor estruturadas aos analistas de segurança e testadores na descoberta de vulnerabilidades futuras. Para isso, foram identificadas as principais plataformas de *Bug Bounty*, onde foi realizado um levantamento das vulnerabilidades com maior ocorrência a partir dos registros dessas plataformas. De posse desse levantamento, foi gerado um conjunto de informações estruturadas e análises a partir de gráficos, com o intuito de apresentar os resultados obtidos. Realizando esse levantamento, identificou-se que as vulnerabilidades mais exploradas foram as de serviços *webs*, em decorrência aos impactos da pandemia.

Palavras-chave: *Bug Bounty Program*. Vulnerabilidades. *Software*.

ABSTRACT

The use of the BBP (Bug Bounty Program) in software companies has become more frequent, since many companies have significantly adapted their approach, integrating more openly vulnerability information obtained externally. Companies like Microsoft and Facebook have developed structured programs in which bug hunters can send the vulnerabilities found and in exchange receive an already predefined reward, depending on the bug the price of the reward can range from 100 to 100,000 Dollars. Threfore, this work aims to carry out a survey and unify the information on certain platforms of Bug Bounty Program, in order to provide better structured information to security analysts and testers in the discovery of future vulnerabilities. For this, the main BugBounty platforms were identified, where a survey of the most frequent vulnerabilities was carried out, based on the records of these platforms. With this survey, a set of structured information and analyzes was generated from graphs in order to present the results obtained. Conducting this survey, it was identified that the most exploited vulnerabilities were the service webs, due to the impacts of the pandemic.

Keywords: Bug Bounty Program. Vulnerabilities. Software.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

ACE	*Arbitrary Code Execution*
API	*Application Programming Interface*
BBP	*Bug Bounty Program*
CEO	*Chief Executive Officer*
DOD	*Department of Defense*
HQL	*Hibernate Query Language*
HTML	*Hypertext Markup Language*
LDAP	*Lightweight Directory Access Protocol*
NoSQL	*Not Only SQL*
OS	*Operation System*
OSI	*Open Source Initiative*
OWASP	*Open Web Application Security Project*
RCE	*Remote Code Execution*
SQL	*Structured Query Language*
URL	*Uniform Resource Locator*
VRP	*Vulnerability Rewards Programs*
XML	*Extensible Markup Language*
XSS	*Cross-Site Scripting*
XXE	*XML External Entities*
ZDI	*Zero Day Initiative*

SUMÁRIO

1 INTRODUÇÃO

Atualmente, existe uma grande diversidade de empresas que utilizam o BBP *(Bug Bounty Program)*, também conhecido como VRP *(Vulnerability Rewards Programs)* ou *Bug Challenges* (BOHME, 2005). Este programa tem o intuito de recompensar pesquisadores de segurança independentes, testadores e *ethical hackers*. Os usuários encontram vulnerabilidades de *softwares* exploráveis e compartilham esse conhecimento com os responsáveis pelo BBP da empresa.

Os operadores de BBP, geralmente representam uma companhia de *software* ou uma empresa que presta serviço terceirizado, no qual é definido o espaço do programa categorizado por tipo de vulnerabilidade, estas inclusas no BBP, tais como, especificações técnicas, critérios de participação, termos, condições, processos de submissão e revisão. O BBP pode fornecer dinheiro ou até mesmo recompensas não monetárias, em alguns casos a recompensa monetária pode chegar até 100.000 dólares (KUEHN, 2014).

Desse modo, considerando os programas de *Bug Bounty* gerado pelas empresas, uma das possibilidades de avaliação é que exista uma maneira de unificar as informações, para que o público que deseja entrar nesse mundo tenha uma maior facilidade de conhecimento. Por isto, este trabalho tem como objetivo geral realizar um levantamento e unificar as informações sobre *Bug Bounty Program*, com o intuito de auxiliar analistas de segurança e *pentesters* na descoberta de vulnerabilidades futuras.

Para que seja atingido o objetivo geral proposto, foi necessário definir os seguintes objetivos específicos:

- Identificar o estado da arte das tecnologias para execução de aplicações em ambientes, a fim de descobrir novas vulnerabilidades.
- Analisar as tecnologias existentes atualmente, com o objetivo de comparar as diferentes alternativas aplicadas pelos testadores.
- Apresentar uma proposta de análise, que permite aos usuários verificar as vulnerabilidades que estão em alta, com a finalidade de auxiliar na descoberta de falhas nas empresas.
- Comparar as vulnerabilidades cadastradas nas plataformas de bugs com os registros disponibilizados por empresas de segurança.
- Mostrar o gasto que as empresas tem em relação a descoberta de vulnerabilidades.

Nesse sentido, o presente trabalho apresenta como principais contribuições:

- Indicação das principais vulnerabilidades atuais presentes nas plataformas, como forma de direcionar possíveis esforços de analistas de segurança e testadores em posteriores buscas por novas vulnerabilidades.
- Realização do cruzamento de informações relacionadas as vulnerabilidades registradas nas plataformas de Bug Bounty selecionadas, como as empresas e os valores pagos pelas mesmas.
- Comparação das vulnerabilidades registradas com dados de relatórios anuais construídos por empresas de *cibersecurity*.

Entende-se que, devido ao registro contínuo de vulnerabilidades nas plataformas selecionadas, este trabalho apresenta um recorte considerando os anos de 2019 e 2020. Da mesma forma, sabe-se que a evolução das técnicas relacionadas às categorias de vulnerabilidades (como no contexto de aplicações web, tem-se a listagem atualizada elaborada pela OWASP *(Open Web Application Security Project)* no seu projeto *Top Ten* (OWASP, 2020)) implica também na necessidade frequente de atualização dos dados. Por outro lado, os resultados apresentados posteriormente neste trabalho reforçam um ordenamento dessas categorias, que se mantém de outros anos anteriores ao recorte aqui apresentado.

O presente trabalho está dividido da seguinte forma: o Capítulo 2 apresenta o referencial teórico que sustenta a base o trabalho, com as explicações sobre os programas de *Bug Bounty* e as principais vulnerabilidades e suas categorias. O Capítulo 3 analisa os trabalhos relacionados com o presente estudo. O Capítulo 4 apresenta a Metodologia para o levantamento e coleta das informações a partir das principais plataformas de *Bug Bounty* selecionadas. Em seguida, o Capítulo 5 mostra a análise dos dados coletados durante o processo de levantamento. Por fim, o Capítulo 6 traz as considerações finais deste trabalho, bem como as limitações e trabalhos futuros provenientes do mesmo.

2 REFERENCIAL TEÓRICO

Neste capítulo são abordadas informações sobre *Bug Bounty Program*, Principais Programas e Principais Vulnerabilidades. O objetivo é fornecer uma visão clara sobre ambos os temas, mostrando a relação entre eles.

2.1 BUG BOUNTY PROGRAM

A história de *bug bounties* inicia nos anos 2000, com o surgimento de programas de divulgação de vulnerabilidades. Um dos exemplos mais notáveis é a ZDI (*Zero Day Initiative*) lançada em 2005, programa ainda ativo, que conta com um modelo de negócios no qual compensa os *hackers* por suas descobertas de vulnerabilidades, e assim auxilia os clientes a desenvolverem correções para os seus produtos.

Nos últimos anos, surgiram inúmeros mercados de vulnerabilidades, consequentemente o número de BBP aumentou significativamente. Empresas de *internet* e *software* começaram a experimentar algumas formas de BBP e outras abordagens, se beneficiando do conhecimento e da análise de técnicos especializados em segurança (KUEHN, 2014).

A ideia de negociar vulnerabilidades de *softwares* e de remunerar caçadores de *bugs* para suas descobertas não é algo novo. *Hackers* têm vendido e comprado vulnerabilidades por um longo tempo em mercados negros, principalmente entre si, e para a sua reputação, quanto mais vulnerabilidades identificadas por um *hacker*, mais conhecido e visto ele fica nesse mercado (CHARLIE, 2007).

Além do lançamento independente de recompensas das empresas, é possível aderir a plataformas de recompensas por *bugs*, como *HackerOne, Cobalt* ou *Bugcrowd*. Estes programas trazem grandes benefícios, um deles é que os serviços são examinados pela grande população de *white hat hackers*, onde seria caro empregar diretamente. Outro motivo, é que devido aos programas serem a grande maioria públicos, as organizações tem uma melhoria contínua na segurança (LASZKA, 2016).

Em maio de 2016 foram encontradas por volta de 20000 vulnerabilidades de segurança e relatadas às respectivas empresas. Em torno de 2500 *white hat hackers* contribuíram nessas descobertas de vulnerabilidades, sendo pago em torno de 7,3 milhões de dólares. Entretanto, com a publicação dos programas há um grande desafio para as empresas, pois como é possível qualquer pessoa contribuir, as empresas acabam recebendo diversos relatórios de baixo valor, surgindo o desafio de analisar cada relatório e verificar se de fato é uma vulnerabilidade ou um falso positivo (LASZKA, 2016).

Quando se refere aos relatórios inválidos, há um número bem significativo, o *HackerOne* por exemplo, divulgou que 54% dos dados enviados foram marcados como inválidos, já a *Bugcrowd*, recebeu 34,5% categorizados como inválidos, ambas analisadas durante o período de Janeiro de 2013 a Junho de 2015 (LASZKA, 2016).

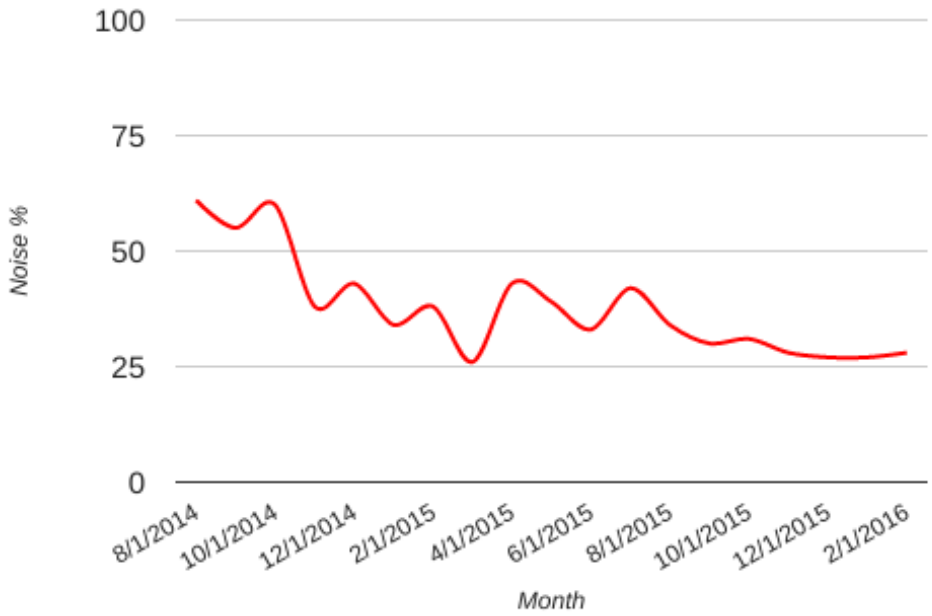
Como solução, as plataformas de recompensas introduziram políticas para realizar a entrega das vulnerabilidades exploradas, com isso as empresas participantes podem reduzir o número de relatórios inválidos, em exemplo disto, o *HackerOne* colocou "Requisitos de Sinal", onde é informado um nível de requisito e conforme varia o nível, apenas *hackers* com histórico comprovado podem enviar relatórios e o "Limitador de Taxas", que possui um controle de relatórios enviados pelos *pentesters*, ou seja, conforme o nível de sinal do *ethical hacker*, maior será a quantidade de envio de relatórios, assim as organizações conseguem aumentar a qualidade de seus relatórios (LASZKA, 2016).

Os Requisitos de Sinal permitem que as empresas definam um limite de sinal, que os *hackers* devem atingir para enviar relatórios a eles, já o Limitador de Taxas fornece aos *hackers* a oportunidade de participar de uma forma limitada, mesmo estando abaixo do requisito de sinal. Se o *hacker* atender ao requisito de sinal, ele pode continuar a participar do programa (HACKERONE, 2016).

Os principais desafios enfrentados pelas empresas, ao executar um programa público em grande escala, são o gerenciamento do ruído ou a grande proporção de relatórios de baixo valor que recebem. O *HackerOne* resolveu se concentrar em melhorar o desempenho do programa público por dois principais motivos. Em primeiro lugar, quanto mais o *HackerOne* permite que os programas sejam continuamente escalados, mais eles geram oportunidades de recompensa para os hackers. Em segundo lugar, os programas que podem ir além dos privados para os públicos, melhoram a defesa das empresas. Esta é uma maneira em que ambas as partes saem ganhando (HACKERONE, 2016).

O time da *HackerOne*, trabalha continuamente para realizar a redução do ruído na plataforma para os clientes. A Figura 1 mostra a melhoria que a plataforma teve entre o ano de Janeiro de 2014 até Janeiro de 2016, com a expansão das melhorias de Reputação e Sinal do *Hacker* feitas nos primeiros 10.000 *bugs* resolvidos no *HackerOne*. O impacto total dos esforços resultou na redução do ruído de toda a plataforma aproximadamente pela metade, durante o período mencionado na Figura 1 (HACKERONE, 2016).

Figura 1: Percentual de Reports Inválidos

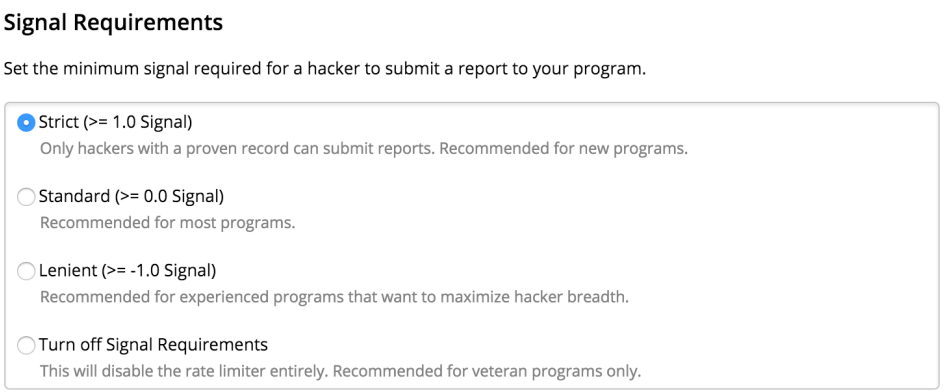


Fonte: HackerOne (2016)

Embora mesmo sendo realizadas as melhorias, o nível de ruído permanece em 28% na plataforma. Esses 28% gerado de todos os relatórios recebidos pelos clientes são classificados como "não aplicável"ou "spam". Realizando a análise das fontes desses relatórios inválidos, a plataforma descobriu que 50% desse ruído é gerado por apenas 10% dos *hackers* na plataforma. Os Requisitos de Sinal e o Limitador de Taxa atualizado serve para reduzir ainda mais o ruído restante (HACKERONE, 2016).

A Figura 2 mostra como as empresas podem definir os requisitos de sinal, assim somente os *hackers* qualificados conforme o requisito podem participar do programa (HACKERONE, 2016).

Figura 2: Requisitos de Sinal



Fonte: HackerOne (2016)

Existem quatro configurações, a empresa pode desligar qualquer limite mínimo de sinal e quando necessário limitar rigidamente a participação do *hacker* com base em sinal alto *"Strict"*. Realizar a configuração *Strict* é a ideal para equipes que preferem poucos relatórios, ou seja só recebem relatórios com qualidade superior ou para as empresas que só podem lidar com um fluxo menor de relatórios. Já diminuir ou desligar os Requisitos de Sinal é a melhor opção para equipes que valorizam ter o número máximo de *hackers* ajudando a encontrar os problemas, porém para isso é necessário ter os recursos para poder gerenciar o volume de relatórios (HACKERONE, 2016).

O limitador de taxa aprimorado funciona em combinação com os requisitos de sinal. É fornecido a todos os *hackers* a oportunidade de participar de um programa, mesmo que o sinal não atenda aos requisitos do programa. Nestes casos, os *hackers* recebem uma série de relatórios de testes, que permitem que, caso seja encontrado algum problema importante, ele possa ser reportado à empresa. O *Rate Limiter* permite que os *hackers* enviem diversos relatórios e alguns casos, é limitado à apenas alguns relatórios por dia, isso varia de acordo com a reputação do *hacker* (HACKERONE, 2016).

A lógica aprimorada do Limitador de taxa é mais ágil, comparando com a maneira que era realizada antes de ser aplicado esses limitadores, é aumentado e diminuído o número de relatórios que pode ser enviado, isso é realizado com base nas mudanças de desempenho do *hacker*. Portanto, conforme os *hackers* melhoram a sua reputação, o número de relatórios que podem ser enviados aumenta (HACKERONE, 2016).

Os novos Requisitos de Sinal e o Limitador de Taxa atualizado, melhoram a qualidade dos relatórios que os BBP podem esperar dos programas públicos. Enquanto isso, permite que todos os *hackers* participem do processo de apresentar as vulnerabilidades importantes identificadas ao conhecimento das empresas (HACKERONE, 2016).

O modelo do BBP revela amplas variações de como os programas podem ser configurados, o BBP pode se concentrar em um produto de *software* único, uma classe de produtos ou a infraestrutura de serviço de uma organização (KUEHN, 2014).

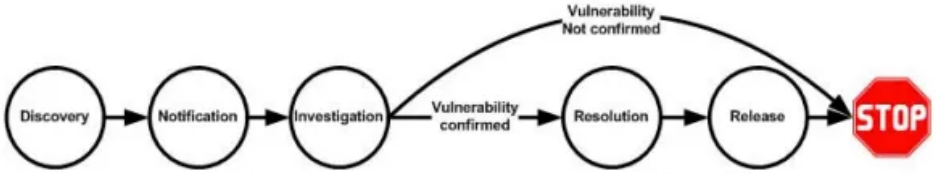
A fim de realizar tarefas de revisão para ajudar desenvolvedores de software e pesquisadores de segurança, foram desenvolvidas técnicas de gerenciamento e abordagens de governança, iniciando pela programação em pares. Também para se proteger de cibercriminosos, empresas contratam *ethical hackers*, devido a possuírem uma mentalidade semelhante ao dos invasores (MAILLART, 2017).

O modelo de divulgação completa evoluiu para divulgação responsável, trata-se de uma prática onde o pesquisador de segurança permite por um período de tempo, a correção da vulnerabilidade antes de realizar a publicação dos detalhes da vulnerabilidade descoberta ao público. Além da melhoria e proteção com as vulnerabilidades encontradas e corrigidas, existe um fator muito importante, o compartilhamento de conhecimento, com isso mais *hackers* se

aprimoram e mais empresas corrigem suas vulnerabilidades (MAILLART, 2017).

Na Figura 3, é apresentado o fluxo de cada vulnerabilidade descoberta até o encerramento do caso, seguindo o fluxo após ser feito a descoberta da vulnerabilidade, é realizado a notificação para empresa, os responsáveis efetuam a investigação do problema apresentado, assim verificam de fato se é uma vulnerabilidade mesmo ou se é um falso positivo, em caso afirmativo é feito a correção e após isso a empresa notifica a *HackerOne* ou outra empresa envolvida no BBP para encerramento do caso, somente com o encerramento do caso o *hacker* recebe a sua recompensa.

Figura 3: Passo a Passo BBP



Fonte: Ramírez (2020)

No ano de 2016 em março, o Pentágono anunciou o seu programa de recompensas, com isso, sendo uma das organizações mais paranoicas do mundo, ofereceu incentivos para que os hackers tentassem invadir seus sistemas e identificassem alguma vulnerabilidade. E mesmo após os programas já terem aumentado muito nos últimos anos, o anúncio feito pelo Pentágono em participar do programa, pode ser considerado um grande avanço para as práticas de segurança cibernética (MAILLART, 2017).

Um *bug* de *software* ou uma vulnerabilidade de *software*, trata-se de uma falha no código, que pode comprometer a segurança do sistema, frequentemente ocorre em *softwares* de computador e protocolos de rede, nos quais estes possuem vulnerabilidades de segurança em consequência não intencionais de escolhas de *design* ou erros matemáticos, por isso é importante observar que um *bug* de *software* não tem características físicas (KUEHN, 2014).

Quando um *bug* é descoberto e possui implicações de segurança em um sistema ou rede, a vulnerabilidade permite que pessoas não autorizadas possam destruir, manipular ou roubar dados do sistema. Para explorar essas vulnerabilidades, o *hacker* precisa escrever o código de *software*, conhecido como *exploit* de *software* (KUEHN, 2014).

2.2 PRINCIPAIS PROGRAMAS

A *Microsoft* lançou seu BBP em junho de 2013, “[...] oferecendo pagamento direto em troca de relatórios de certos tipos de vulnerabilidades e técnicas de exploração "(MICROSOFT, 2013b). O BBP inicial consistiu em três componentes que incentivam a descoberta de *bugs* de segurança, mas acabou encorajando ainda mais a apresentação de novas técnicas de exploração. Em novembro de 2013, a *Microsoft* estendeu o programa (MICROSOFT, 2013a).

Anteriormente, o BBP era limitado aos especialistas em segurança que projetaram novas técnicas de *by-pass* sem precedentes, as quais eram observadas em ataques reais. Consequentemente, os participantes eram desde peritos de segurança a milhares de indivíduos qualificados (MICROSOFT, 2013a).

O *Facebook* iniciou seu programa com recompensas monetárias em julho de 2011, os *bugs* válidos encontrados valiam 500 dólares no mínimo, seus *submitters* foram nomeados no portal de *White Hat* do *Facebook*, três semanas após seu início, o *Facebook* informou que mais de 40.000 dólares já haviam sido pagos (FACEBOOK, 2011). Um ano depois, o *Facebook* ampliou o programa e incluiu a sua infraestrutura interna, abrangendo as redes corporativas e a infraestrutura de produção (ROBERTSON, 2012).

Em 2013, o *Facebook* informou que recebeu 14.763 envios, dos quais 687 foram falhas de segurança elegíveis, para as quais 1,5 milhão de dólares foi pago a 330 pesquisadores. Foram categorizados 41 *bugs* como alta severidade, a maior recompensa do *Facebook* atingiu 33.500 dólares.

A *Netflix* com o objetivo de manter a segurança dos seus mais de 117 milhões de usuários, lançou em 2013 o seu programa de *Bug Bounty* através da plataforma *Bugcrowd*, oferecendo recompensas entre 100 a 15 mil dólares por *bug*, levando em consideração sempre o nível de prioridade estabelecido pela empresa (HARÁN, 2018).

Desde o lançamento do programa, conforme descrito por (HARÁN, 2018) já haviam resolvido 190 problemas válidos. Em 2018 foi aumentado o escopo do programa, bem como o número de pesquisadores, com isso foram feitas melhorias para motivar os pesquisadores a participar, utilizando prioridades de acordo com o nível de *bug*, um tempo de resposta mais rápido e com maior interação com os pesquisadores.

A *HackerOne* fundada em 2012, vem conectando empresas com pesquisadores de segurança para auxiliar a encontrar vulnerabilidades de *softwares*. Até então a plataforma vem sendo utilizada por muitas empresas para recompensas públicas e privadas, como *Adobe, Kaspersky, Twitter, Microsoft, Facebook e Rockstar Games* (SPRING, 2017).

Em 2017 a empresa anunciou uma versão gratuita da sua plataforma de *bug bounty*, denominando *HackerOne Community Edition*, responsável pelo fornecimento de ferramentas para projetos *open-source*, gerenciando cumprimento de vulnerabilidades e assim criar

programas de recompensa com a finalidade de melhorar a segurança dos *softwares* (SPRING, 2017).

Para um projeto se qualificar para o serviço *Community Edition*, devem estar ativos há pelos menos três meses e cobertos por uma licença OSI *(Open Source Initiative)*, na qual essa licença permite que o *software* seja utilizado, modificado e compartilhado livremente (SPRING, 2017).

A própria empresa reconhece que códigos *open-source* sustentam muitos produtos e serviços, alegando que foi obrigada a oferecer a assinatura do *HackerOne Professional* gratuitamente. Segundo a *HackerOne*, em 2017 haviam 36 projetos de código aberto utilizando a plataforma, mais de 1.200 vulnerabilidades já haviam sido corrigidas nos projetos, incluindo *Ruby, Rails, Discourse, Django, GitLab, Brave e Sentry* (SPRING, 2017).

No ano de 2016 a *Kaspersky* lançou seu programa público de recompensas de *bugs*, em parceria com a *HackerOne*, uma das primeiras plataformas a oferecer recompensas pela descoberta de *bugs*, já em 2018, a empresa fez a atualização de seu programa de acordo com a *Global Transparency Initiative* (KASPERSKY, 2020).

As ameaças virtuais vem crescendo cada vez mais, a cada momento vão se tornando mais complexas, com isso exigindo a identificação e implementação contínua, utilizando ferramentas eficazes que oferecem um nível mais desenvolvido de proteção. Os BBP funcionam como uma medida de segurança eficiente e comprovada, onde incentiva pesquisadores independentes a achar e publicar vulnerabilidades de *softwares* às empresas de maneira segura, com isso as empresas são capazes de corrigir as vulnerabilidades sem trazer riscos aos clientes (TIINSIDE, 2016).

A *Karspersky Lab* no decorrer do tempo ampliou o programa de recompensas, incluindo prêmios de até 100 mil dólares pela descoberta de vulnerabilidades. Qualquer membro da plataforma *HackerOne* tem a oportunidade de ganhar essa recompensa, a inclusão dos novos prêmios são 20 vezes maiores do que os que já possuíam (TIINSIDE, 2016).

O CEO *(Chief Executive Officer)* Eugene Kaspersky, disse que "descobrir e corrigir bugs é prioridade para a empresa. Convidamos os pesquisadores de segurança a nos ajudarem a garantir que não exista vulnerabilidades em nossos produtos. A imunidade de nosso código e os níveis mais altos de proteção que oferecemos aos clientes são os princípios básicos de nossa empresa e um pilar fundamental de nossa Iniciativa de Transparência Global"(ABES, 2018).

O *Google* pagou em terno de 2,9 milhões em recompensas no ano de 2017, normalmente as recompensas variam entre 500 e 100.000 dólares, dependendo do tipo de vulnerabilidade e o tempo gasto. Atualmente existem vários programas, incluindo *Vulnerability Research Grants Program e Patch Rewards Program*, esses programas são internos do Google, o primeiro pagou um total de 125.000 dólares a 50 pesquisadores no ano de 2017, enquanto o outro 50.000 para melhor segurança no *software open-source* (ABES, 2018).

O *Bugcrowd* é uma plataforma web que oferece uma estrutura para que as empresas deixem seu software, afim de serem testados por pesquisadores, essa plataforma atua para encontrar e receber informações sobre testes de penetração, sendo em execução e até oferecendo suporte na avaliação de envios por ferramentas de análise (SHULZ, 2014).

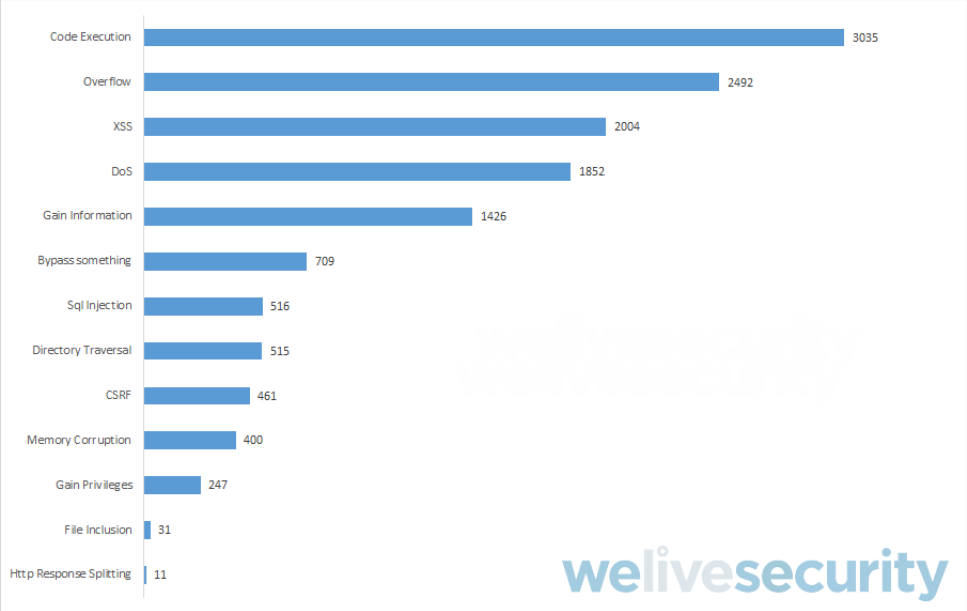
A *Apple* conhecida como uma das empresas de tecnologia que possui um programa de recompensas mais desvalorizado, não preza pela descoberta de vulnerabilidades externamente, oferecendo um valor de recompensas baixo, comparado ao de suas concorrentes, também oferecendo poucos recursos em relação às ferramentas para a descoberta desses *bugs* (SANTANA, 2019).

Esta visão da empresa mudou a partir do ano de 2019, iniciando uma expansão significativa no programa de recompensas e nas pesquisas de *software* da empresa, com isso o programa ficou aberto a todos os pesquisadores interessados e não mais apenas para os desenvolvedores de iOS registrados, além de começar a oferecer ferramentas avançadas e versões especiais do iPhone, para que os profissionais pudessem fazer suas análises (SANTANA, 2019).

2.3 PRINCIPAIS VULNERABILIDADES

Na Figura 4 estão listados os casos reportados em 2018 de vulnerabilidades de acordo com o site *WeliveSecurity*.

Figura 4: Lista de Vulnerabilidades em 2018



Fonte: BARBOSA (2019)

As principais vulnerabilidades de segurança em aplicações web são as mais comuns, e com isso acabam sendo as mais perigosas, são por elas que os hackers normalmente começam a explorar as vulnerabilidades. A seguir nos próximos capítulos, serão apresentadas as TOP 10 divulgadas no OWASP (2020) e algumas das vulnerabilidades que mais foram divulgadas no *HackerOne* e *BugCrowd*, conforme o levantamento realizado.

2.3.1 *Code Execution*

É também conhecido como ACE *(Arbitrary Code Execution)*, essa vulnerabilidade permite a execução de comandos ou linhas de código em um processo, mesmo não fazendo parte da estrutura original do processo. Essa é uma vulnerabilidade que está no topo do *ranking*, pois todo *software* que não é desenvolvido adequadamente, pode conter essa vulnerabilidade de execução de código (BARBOSA, 2019).

RCE (*Remote Code Execution*) é a capacidade de explorar a vulnerabilidade e *softwares* remotamente, caso o atacante consiga explorá-la em serviços ou em protocolos amplamente utilizados, pode acabar comprometendo diversos computadores simultaneamente, como por exemplo, quando ocorreu a falha do protocolo SMBv1, permitindo a propagação do *WannaCry* em 2017 (BARBOSA, 2019).

2.3.2 *Injection*

É um ataque de injeção que facilita a execução de instruções, como SQL (*Structured Query Language*), NoSQL (*Not Only SQL*, OS (*Operation System*) e LDAP (*Lightweight Directory Access Protocol*), ocasionadas quando dados não confiáveis são enviados para um intérprete como parte de um comando ou consulta (OWASP, 2020). Os dados do invasor podem induzir um intermediário a realizar comandos não intencionais ou acessar dados sem autorização, podendo adicionar ou modificar as consultas existentes através de uma URL (*Uniform Resource Locator*) ou banco de dados.

Com isso, é possível controlar o servidor de banco de dados atrás de aplicações web, utilizando vulnerabilidades de injeção SQL para ignorar as medidas de segurança do aplicativo. Na maioria dos casos, os invasores realizam adição, edição ou exclusão dos dados, sendo possível roubar informações dos clientes, dados pessoais, segredos comerciais, entre outros (MOHAN, 2019).

2.3.3 *Cross-Site Scripting (XSS)*

Cross-Site Scripting (XSS), assim como SQL Injection, trata-se de um ataque de injeção de código, afetando tanto o cliente quanto o navegador. Esses invasores conseguem inserir códigos maliciosos de *javascript* e *tags html* na página, e na maioria dos casos, o XSS requer um determinado comportamento do usuário, como acessar um *link* específico. Quando bem executada, este tipo de vulnerabilidade é muito difícil de ser identificada, principalmente para usuários leigos (MOHAN, 2019).

Os erros do XSS ocorrem sempre que um aplicativo inserir dados não confiáveis em uma nova página da *Web* sem a validação necessária, ou atualizar uma página da *Web* já existente, com informações fornecidas pelo cliente, usando uma API do navegador, que pode gerar um HTML ou *JavaScript* (OWASP, 2020).

Através do XSS, os invasores conseguem executar *scripts* no navegador da vítima, e estes permitem que sejam capturadas sessões do usuário, com intuito de corromper sites ou até mesmo redirecionando o usuário para sites maliciosos (OWASP, 2020).

2.3.4 *Broken Authentication*

Constantemente, as funções de aplicativo referentes à autenticação e o gerenciamento de sessões são implementadas incorretamente, possibilitando que os invasores comprometam senhas, chaves ou *tokens* de sessão ou descubram outras falhas de implementação, com a intenção de assumir a identidade de outros usuários de forma temporária ou permanentemente.

2.3.5 *Sensitive Data Exposure*

Atualmente, os dados confidenciais não possuem a segurança necessária, garantida por diversos aplicativos da *Web e APIs*, como financeiro, assistência médica e PII (OWASP, 2020). A partir dessa situação, o invasor consegue roubar ou alterar essas informações protegidas minimamente, realizando assim fraudes no cartão de crédito, roubo de identidade ou demais crimes. Estes dados confidenciais podem ser comprometidos sem proteção extra, como criptografia em repouso ou em trânsito, e demandam precauções especiais sempre que o navegador for mudado (OWASP, 2020).

2.3.6 *XML External Entities (XXE)*

As entidades externas dispostas em documentos XML são utilizadas como referência para muitos processadores XML mais antigos ou mal configurados. Entidades externas podem ser aproveitadas para divulgar arquivos internos, usando o manipulador de URI de arquivos, compartilhamentos de arquivos internos, varredura de portas internas, execução remota de código e ataques de negação de serviço.

2.3.7 *Broken Access Control*

Controle de acesso quebrado. As restrições sobre o que os usuários autenticados têm permissão para fazer, geralmente não são aplicadas corretamente. Os invasores podem explorar essas falhas para acessar funcionalidades ou dados não autorizados, como acessar contas de outros usuários, visualizar arquivos confidenciais, modificar dados de outros usuários, alterar direitos de acesso e etc.

2.3.8 *Security Misconfiguration*

A configuração incorreta da segurança é o problema mais comum. Isso geralmente resulta de configurações padrão inseguras, incompletas ou *ad hoc*, armazenamento em nuvem aberta, cabeçalhos HTTP configurados incorretamente e mensagens de erro detalhadas que contêm informações confidenciais. Não apenas todos os sistemas operacionais, estruturas, bibliotecas e aplicativos devem ser configurados com segurança, mas devem ser corrigidos e atualizados em tempo hábil.

2.3.9 *Insecure Deserialization*

A desserialização insegura geralmente leva à execução remota de código. Mesmo que as falhas de desserialização não resultem em execução remota de código, elas podem ser usadas para executar ataques, incluindo ataques de repetição, ataques de injeção e ataques de escalonamento de privilégios.

2.3.10 *Using Components with Known Vulnerabilities*

Componentes, como bibliotecas, estruturas e outros módulos de *software*, são executados com os mesmos privilégios que o aplicativo. Se um componente vulnerável for explorado, esse ataque poderá facilitar a perda séria de dados ou a aquisição de servidores. Aplicativos e APIs que usam componentes com vulnerabilidades conhecidas podem minar as defesas de aplicativos e permitir vários ataques e impactos.

2.3.11 *Insufficient Logging & Monitoring*

O registro e o monitoramento insuficientes, juntamente com a integração ausente ou ineficaz com a resposta a incidentes, permitem que os invasores continuem atacando os sistemas, mantenham a persistência, façam o giro para mais sistemas e violem, extraiam ou destruam dados. A maioria dos estudos de violação mostra que o tempo para detectar uma violação é superior a 200 dias, geralmente detectados por partes externas, em vez de processos ou monitoramento interno.

3 TRABALHOS RELACIONADOS

Este capítulo apresenta os trabalhos relacionados com o presente estudo, de forma a entender quais são os aspectos e características envolvidos nas pesquisas sobre o assunto e, a partir disso, promover um entendimento base sobre o estado-da-arte do tema.

Inicialmente, o trabalho proposto por Hata (2017), faz uma análise do crescimento do BBP, tanto da participação de empresas como de projetos de código aberto, porém como é citado, pouco sabe-se sobre as características dos contribuintes do programa de recompensa. O objetivo da pesquisa é propor uma análise sobre o entendimento desse público e destacar a heterogeneidade entre eles. No presente trabalho, além de trazer a análise do crescimento de vulnerabilidades no ano de 2019 e 2020, também é apresentado os tipos de vulnerabilidades que mais se destacaram entre as 500 analisadas no relatório, já no trabalho desenvolvido por Hata (2017), o foco foi apresentar as características dos contribuintes.

Já Canfield (2015), propôs um modelo centralizado gerenciado por um negociador autorizado, para atuar em nome da sociedade e trazer mais clareza no processo de recompensa de bugs, cuja proposta visa incentivar a transparência, seja ela implementando uma divulgação organizada de *software*, para que os serviços e interesses de segurança nacional se baseiem para que tenha o incentivo dos fornecedores a produção de *patches* mais rapidamente, então o autor pretende seguir na solução dos problemas, já na pesquisa realizada o objetivo é incentivar o público a ingressar nesse mercado, contribuindo com o apoio de busca de novas vulnerabilidades.

No estudo proposto por Kuehn (2014), os autores abordam a teoria econômica institucional, com o intuito de examinar os desenvolvimentos recentes dos programas de recompensas por *bugs*. O estudo destaca que as empresas como *Microsoft*, *Adobe* e *Oracle* receberam considerável atenção da mídia devido a problemas graves de segurança, a partir disso, elas têm incentivos para corrigir *bugs* no *software*, uma vez que estes são descobertos. Nesse sentido, os autores discutem a forma de divulgação das vulnerabilidades descobertas pelos profissionais, abordando que existem questões controversas na comunidade de segurança da informação, já que pesquisadores de segurança podem enfrentar desafios legais quando compartilharam suas descobertas com o fornecedor do *software* ou divulgaram essas informações ao público.

O trabalho argumenta, que os programas de recompensas por *bugs* constituem uma mudança significativa na maneira como as informações de vulnerabilidade são sistematicamente adquiridas pelos fornecedores de *software*. Dessa forma, os autores afirmam que normas e práticas emergentes reduzem o nível de incerteza na troca de informações críticas sobre vulnerabilidades. Para validar seu estudo, Kuehn (2014) fornecem uma narrativa histórica do desenvolvimento de programas de recompensas por *bugs* e suas mudanças nas práticas de

segurança relacionadas, além de fornecer uma análise comparativa de vários programas de recompensa. A partir dessa problemática que o estudo de Kuehn (2014) faz uma análise de perspectiva institucional, para explicar o surgimento de programas de recompensas através de uma pesquisa empírica, baseada em uma análise de documentos. Assim, o estudo busca contribuir para as discussões sobre a divulgação de vulnerabilidades de *software* e sobre políticas envolvendo a regulamentação das mesmas.

Huang (2016), em sua pesquisa, apresenta uma discussão sobre a eficácia de programas de recompensa de *bugs* ao atrair *hackers* externos, para encontrar e divulgar possíveis falhas de uma maneira responsável. Nesse sentido, os autores abordam a existência de muitos programas de recompensas diferentes e como isso pode atuar na diversidade e concentração dos participantes, para a construção de sua reputação no ecossistema de descoberta de vulnerabilidades.

A partir disso, o trabalho de Huang (2016) propõe uma metodologia para entender como os *hackers* distribuem sua atenção para o trabalho, envolvendo diferentes programas e demonstram como resultado, essa relação entre diversidade e concentração, sugerindo uma estratégia eficaz para a atuação alternada dos participantes.

O trabalho desenvolvido é relacionado ao Huang (2016), onde é mostrado as vulnerabilidades que mais ocorrem nos tempos atuais e como explorá-las, assim o público que possui interesse em participar da realização de *exploits*, afim de identificar as vulnerabilidades, terá uma maneira ao qual poderá seguir.

A pesquisa de Walshe (2020) demonstra, inicialmente, a importância dos programas de recompensa de *bugs* como principal local para realização das tarefas de identificar vulnerabilidades por parte de *hackers*. O estudo apresenta resultados de uma análise empírica, realizada com os dados disponíveis em duas plataformas de recompensas de *bugs* para entender os custos e benefícios dos programas de recompensas de *bugs*, tanto para os participantes quanto para as organizações.

Essencialmente, esses resultados são voltados para os aspectos econômicos dos programas de recompensas por *bugs*, já que as contribuições estão diretamente relacionadas com as comparações de custo médio de operação. Nesse sentido, o trabalho de Walshe (2020), assim como o presente trabalho, analisa os dados de plataformas de *bug bounty* de maior expressão nesse mercado. Contudo, devido a exclusividade de viés econômico da análise proposta por Walshe (2020), propõe-se neste estudo trazer dados direcionados as classificações de vulnerabilidades como forma de avanço à pesquisa do autor.

Bienz (2020), apresenta na sua análise a diferença entre os tipos de *bugs*, afim de auxiliar os desenvolvedores de *software* a identificarem qual o melhor o benefício em um *Bug Bounty Program*. Desenvolvedores de *software* utilizam BBP no qual premiam a detecção de *bugs* em seu *software*. O BBP permite que os desenvolvedores discriminem perfeitamente entre

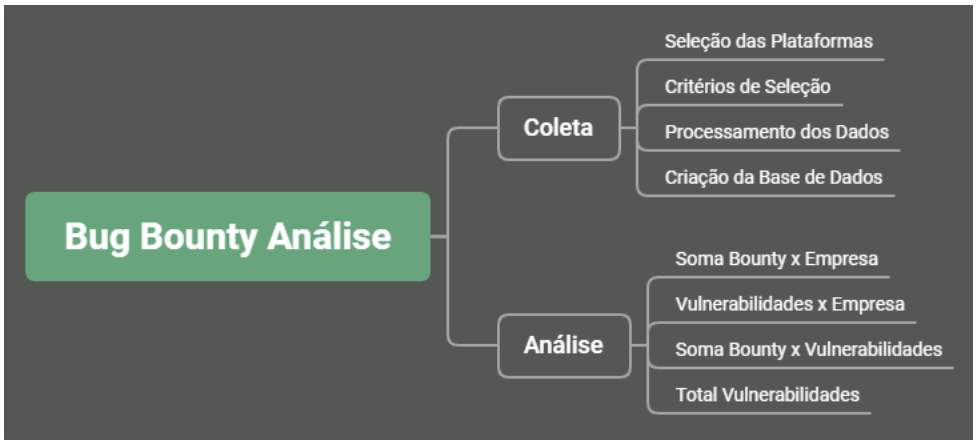
diferentes tipos de *bugs*, além de auxiliar a evitar os custos de reputação dos *bugs* explorados. Por fim, conclui-se que os benefícios BBP não depende apenas das características do *software* subentendidos, mas também que um BBP relaciona-se crucialmente com outros elementos da estratégia de segurança.

Nos trabalhos dos autores mencionados acima, cada um focou em uma maneira diferente de apresentar ao mercado e ao público, a visão de como ingressar nos programas e das empresas a participarem do BBP, Hata (2017) foca no crescimento de vulnerabilidades no BBP, já Kuehn (2014) abrange as vulnerabilidades a fim de terem uma ação mais rápida nas correções dessas vulnerabilidades. Huang (2016) demonstra a eficácia dos BBPs buscando de atrair os *hackers* externos para auxiliar na descoberta, Walshe (2020) demonstra como no presente trabalho o gasto realizado pelas empresas para recompensar as vulnerabilidades identificadas. Bienz (2020) apresenta a diferença entre os tipos de *bugs*, com o intuito de auxiliar o time de desenvolvimento de *software* a identificar o melhor benefício no BBP. Todos os trabalhos relacionados de uma forma ou de outra, demostram a eficácia que é uma empresa ingressar no BBP e de *ethical hackers*, contribuírem na identificação das vulnerabilidades.

4 METODOLOGIA

Neste conteúdo são abordadas as ações realizadas para efetuar a coleta dos dados, diante disso, foi realizada uma classificação dos BBP coletados utilizando alguns critérios, como por exemplo, o período analisado e o retorno monetário, após essa classificação, foi efetuada a análise dos dados obtidos durante a coleta.

Figura 5: Etapas de Análise *Bug Bounty*



Fonte: ELABORADO PELO AUTOR

4.1 SELEÇÃO DAS PLATAFORMAS

Para o início do processo contido na metodologia, foram investigadas, através do referencial teórico e de pesquisa empírica, quais seriam as plataformas alvo para a execução deste trabalho. Conforme os estudos apresentados HackerOne (2020), Apple (2020), BugCrowd (2020) e Google (2020), verificou-se que a plataforma *HackerOne* - descrita previamente no Capítulo 2 - detém a maior parte da atenção e foco entre os programas existentes.

Para não destinar os esforços somente a uma plataforma, estabeleceu-se o requisito de complementar o levantamento deste estudo com, no mínimo, a adição de outra plataforma para fornecer os dados. Nesse sentido, buscou-se de forma similar ao processo anteriormente comentado definir outra plataforma. Assim, entende-se que, devido as funcionalidades e estruturas das aplicações, a plataforma *BugCrowd* mostrou-se relevante para o trabalho.

Figura 6: Plataformas Analisadas

Plataforma	Publicação Visivel	Cadastro	Recompensas	Restrição de Software
HackerOne	X		X	
Apple		X	X	X
Bugcrowd	X	X	X	
Google		X	X	

Fonte: ELABORADO PELO AUTOR

Após essa seleção das plataformas *HackerOne* e *BugCrowd*, foi verificado que estas possuem mais detalhes dos programas de recompensas, além de utilizar internamente o BBP, estas prestam o serviço terceirizado para outras empresas, a grande maioria das empresas não possuem uma base de dados com as vulnerabilidades encontradas, somente detalhamento do que deve se explorado.

4.2 CRITÉRIOS DE SELEÇÃO

Para estabelecer quais as vulnerabilidades poderiam ser selecionadas no levantamento, foi realizado uma análise das vulnerabilidades do período entre o ano de 2019 e o presente momento do ano de 2020, já que pesquisas com dados mais antigos podem não trazer resultados satisfatórios, tal como vulnerabilidades que não são mais exploráveis. A pesquisa executada com informações recentes tendem a trazer os dados que estão mais em alta no mercado, devido a tecnologia evoluir constantemente, as vulnerabilidades exploráveis vão mudando, em consequência disso as empresas vão se adaptando e se protegendo dos *bugs* mais comuns.

Foi coletado somente vulnerabilidades que tiveram recompensas e que a publicação estava liberada, há vulnerabilidades divulgadas que não possuem liberação ao publico, somente a equipe terceira e a empresa que está com a falha conseguem visualizar todas as informações, vulnerabilidades que não foram corrigidas não estão visíveis para o público.

Os critérios escolhidos para realizar a coleta de dados, além de selecionar somente vulnerabilidades que tiveram recompensas, foi analisar o período de janeiro de 2019 até o mês de agosto de 2020. Para complementar a análise, a fim de ter uma coleta mais detalhada de dados, foram filtrados apenas os casos que continham os dados completos, ou seja, que possuíam dados como: titulo, valor da recompensa, tipo de vulnerabilidade, informação de qual serviço era afetado pela vulnerabilidade *(mobile, web,* aplicação) e a empresa que possui a vulnerabilidade explorada.

Outro item importante avaliado e considerado na escolha dos *bounties* foi o sumário, buscando aqueles que apresentavam informações sobre o impacto que ocorria na vulnerabilidade explorada, além de possuir também uma breve descrição de como reproduzir a mesma, somente assim as equipes técnicas conseguem reproduzir os testes.

De forma resumida, a partir das explicações acima, os critérios de seleção de vulnerabilidades são os seguintes:

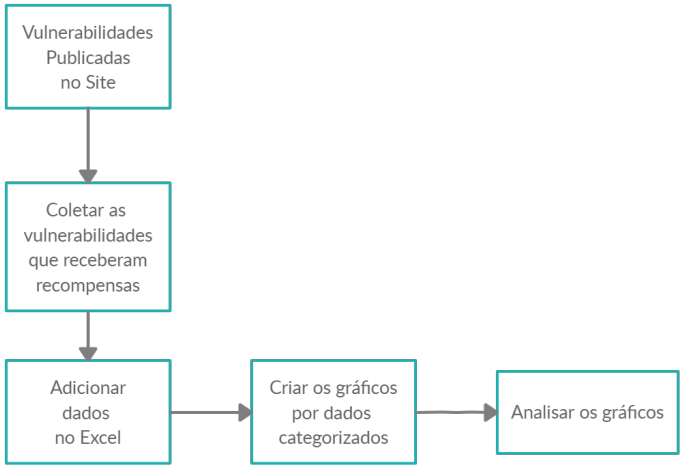
- Status. Necessidade do status, pois somente vulnerabilidades finalizadas são divulgas.
- Período de Publicação. Somente fazem parte vulnerabilidades entre o período de Janeiro de 2019 até Agosto de 2020.
- Necessidade de Recompensa relacionada.
- Apresentar o detalhamento da descoberta.
- Descrição do tipo de vulnerabilidade
- Cenário de aplicação. Mobile, web

4.3 PROCESSAMENTO DOS DADOS

Os dados foram coletados dos sites *HackerOne* e *BugCrowd*, foi realizado a criação de uma base de dados em um arquivo *excel*, foram catalogados vulnerabilidades exploradas no período de Janeiro de 2019 até Agosto de 2020, somente vulnerabilidades que receberam recompensas em valor monetário foram inseridas e que possuíam no mínimo as informações abaixo.

- Titulo
- *Status*
- Data Divulgação
- Para qual empresa foi reportado
- Sistema Divulgado
- Vulnerabilidade
- Valor da Recompensa
- *Hacker* que descobriu a vulnerabilidade
- Breve Descrição da Vulnerabilidade Reportada

Figura 7: Fluxo da Coleta Realizada



Fonte: ELABORADO PELO AUTOR

4.4 CRIAÇÃO DA BASE DE DADOS

Para realizar a criação da base de dados, os dados foram classificados e processados utilizando os critérios a seguir, coletar somente as vulnerabilidades que já estavam finalizadas, somente as que receberam recompensas e que estavam com os dados de tipo de vulnerabilidade, empresa, titulo e com o fechamento após 2019. Após a coleta de dados, foram realizadas classificações organizando por valor total de *bounty* por empresa, valor total por vulnerabilidade, total de vulnerabilidades.

5 RESULTADOS E ANÁLISE

Neste capítulo são apresentados os resultados e a análise a partir dos dados obtidos pela metodologia anteriormente descrita.

Inicialmente, é interessante abordar o fato de que, de acordo com o *site* da *HackerOne*, no período disposto nos critérios de seleção apresentados anteriormente, foram cadastradas 7.168 vulnerabilidades, sendo em 2019 o total de 4.716 *bugs*, e nos primeiros 4 meses de 2020 já foram publicados 2452. Todos estes recompensados para os *hackers*.

Do total de vulnerabilidades cadastradas, foram coletados 500 *bugs*, como intuito de buscar quais as vulnerabilidades de maior ocorrência no último ano, sendo este um número suficiente para ter uma análise mais detalhada. Devido as mesmas vulnerabilidades se manterem sempre no topo, quanto mais dados eram inseridos, os cinco *bugs* que mais ocorriam, permaneciam sempre os mesmos.

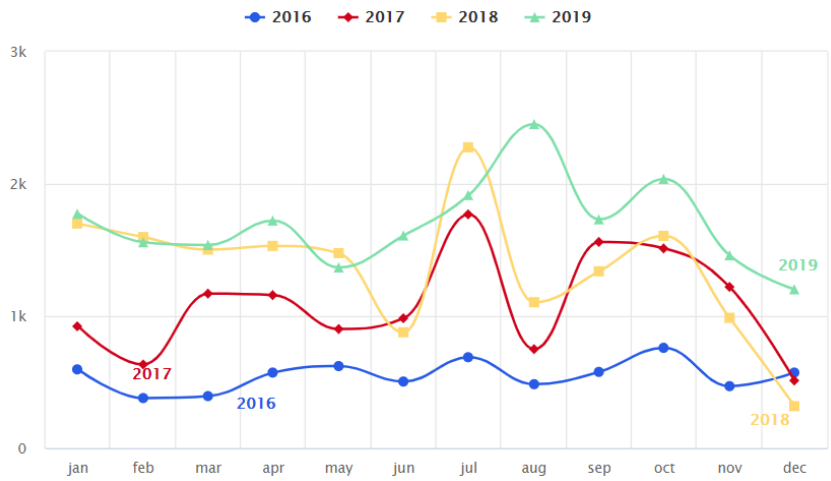
Das vulnerabilidades coletadas foi feito o levantamento de quais empresas tiveram vulnerabilidades exploradas e a quantia total que cada empresa teve que pagar nas recompensas, possibilitando identificar as que mais possuem problemas com os bugs. Não necessariamente a empresa que pagou mais nas recompensas, foram as empresas que mais tiveram vulnerabilidades cadastradas, ou seja, a empresa que mais gastou, teve poucos bugs, porém com valores mais elevados.

Outro ponto importante a destacar, é que nem sempre as vulnerabilidades mais exploradas são as que somam um valor maior, pois quanto mais frequente o *bug* aparecer, menos valorizado ele será. Normalmente as que possuem um número menor, são as mais valorizadas, devido serem as que menos ocorrem e as que podem afetar drasticamente a empresa. Como exemplo disto, podemos citar os sites de *ecommerce*, nos quais sofrem bastante com o XSS, por trazer algum *redirect* para algum *site* malicioso, podendo ser afetado com o roubo de dados, acessos, contas e etc.

Na Figura 8 está listado um comparativo das vulnerabilidades publicadas mensalmente nos últimos 4 anos, conforme descrito por (BEKERMAN, 2020) o número geral de novas vulnerabilidades em 2019 (20.362) aumentou 17,6% em comparação a 2018 (17.308) e 44,5% em comparação a 2017 (14.086).

De acordo com os dados coletados pela *Imperva*, 47% das vulnerabilidades possuem uma exploração pública disponível para *hackers*. Além disso 40,2% das vulnerabilidades não possuem uma solução disponível, tal como atualização de *software*, soluções alternativas ou até mesmo correção de *software* (BEKERMAN, 2020).

Figura 8: Número de Vulnerabilidades

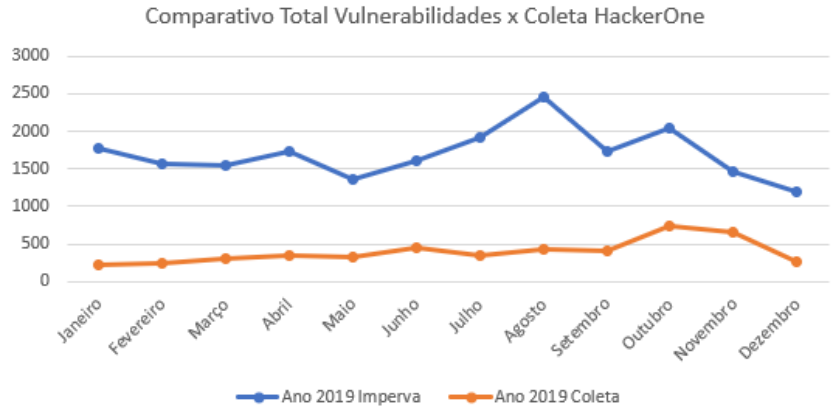


Fonte: BEKERMAN (2020)

Na Figura 9, está sendo exibido um comparativo do ano de 2019, entre os dados coletados pelo time da *Imperva* e dos sites *HackerOne* e *BugCrowd*. Com base nos dados obtidos, observa-se que atualmente só é publicado em torno de 20% das vulnerabilidades, através de uma mão de obra terceirizada, onde as mesmas foram pagas pelas respectivas empresas que tiveram a vulnerabilidade explorada.

Os demais *sites* não mantém uma base de vulnerabilidades armazenadas para consulta, somente está publicado como funciona o BBP e os valores das recompensas ou em alguns casos por exemplo na Apple (2020) é necessário realizar um cadastro e mesmo assim não é visivelmente as vulnerabilidades já exploradas.

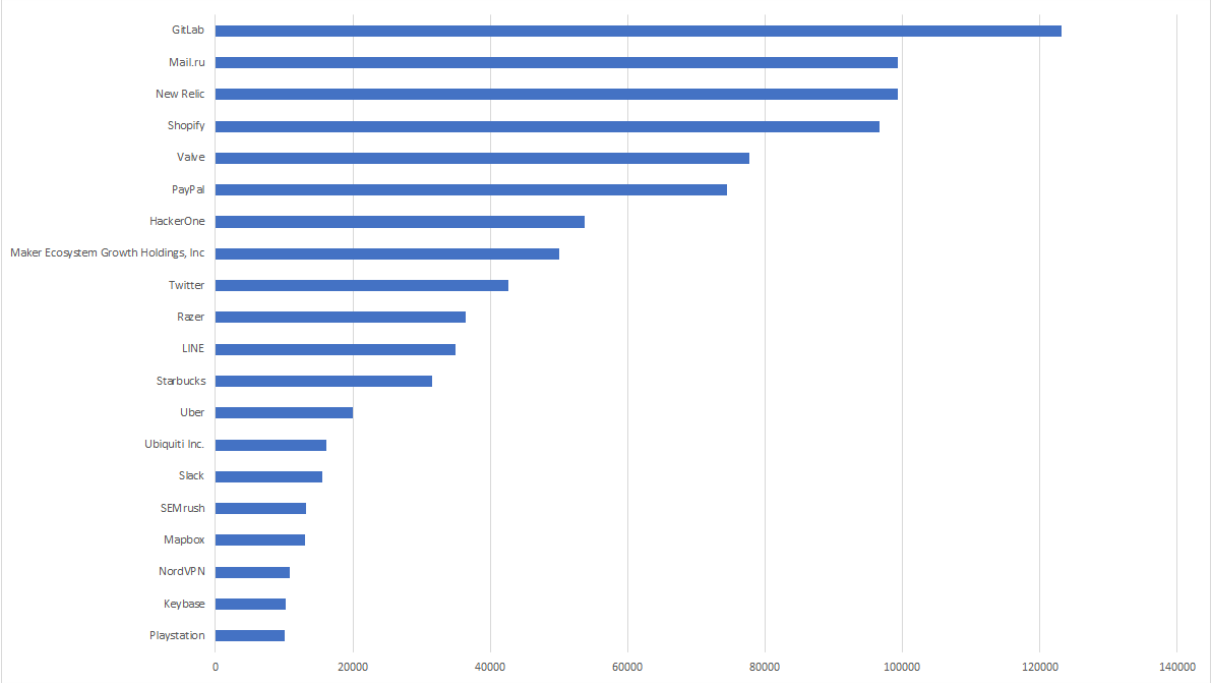
Figura 9: Comparativo Vulnerabilidades



Fonte: ELABORADO PELO AUTOR

5.1 ANÁLISE DE BOUNTY POR EMPRESA

Figura 10: Soma Bounty x Empresa



Fonte: ELABORADO PELO AUTOR

O gráfico na Figura 10, mostra a soma total de recompensas pagas no período de Janeiro de 2019 até Agosto de 2020, separando por valores pagos por cada empresa. É possível identificar a soma gasta por cada empresa e notar que empresas de *ecommerce*, jogos *online*, gerenciador de contas de *e-mail* e repositório de *softwares* tem um valor maior de recompensas.

Atualmente a tendência é a realização de diversas ações via internet, uma maior praticidade para execução e não é necessário se locomover de um lugar para outro, porém com isso fica propício a pessoas mal intencionadas e explorarem vulnerabilidades nesses meios, como por exemplo, compras online, pagamento de contas e outros serviços, a vulnerabilidade de XSS fica mais favorável a ser exploradas nesses sites.

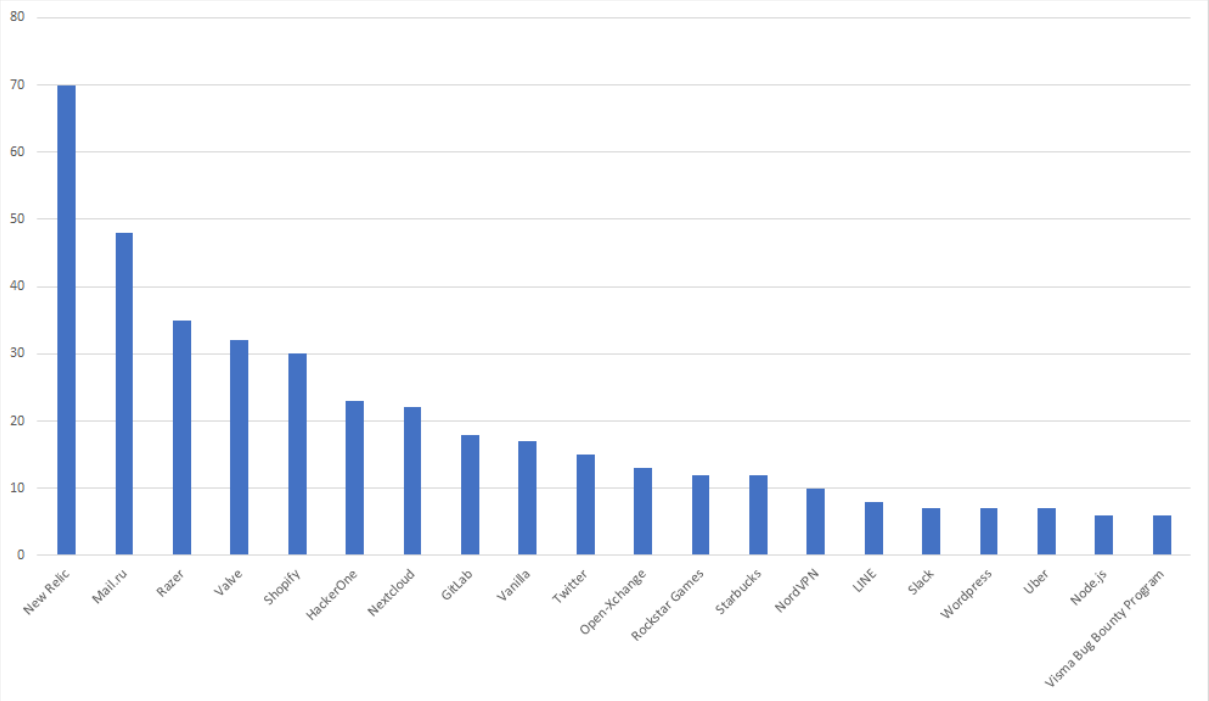
5.2 ANÁLISE DE VULNERABILIDADES POR EMPRESA

Comparando a Figura 11 com a anterior, já podemos notar que mesmo empresas pagando um valor alto nas recompensas, nem sempre estarão no topo do total de vulnerabilidades exploradas, devido a cada vulnerabilidade ter uma classificação por nível sendo (alto, crítico, médio e baixo), também se deve analisar a criticidade da vulnerabilidade explorada pelos atacantes.

Atacantes preferem explorar vulnerabilidades que normalmente ocorrem em determinado tipo de site ou serviço e assim, por exemplo sites de *ecommerce* tendem a ter vulnerabilidades exploradas como XSS, *SQL Injection* entre outras. Assim os atacantes iniciam pelas análises básicas ao invés de explorar as de alta criticidade, com isso *sites* pequenos tendem a serem mais explorados.

Na Figura 11 a empresa que mais teve vulnerabilidades exploradas foi a *New Relic*, trata-se de uma empresa que desenvolve software baseado em nuvem, ou seja tem seus dados com maior chance de exposição e devido a nuvem ser um serviço novo e com seu crescimento constante, tende a ter mais chances de ter vulnerabilidades.

Figura 11: Total Vulnerabilidades x Empresa

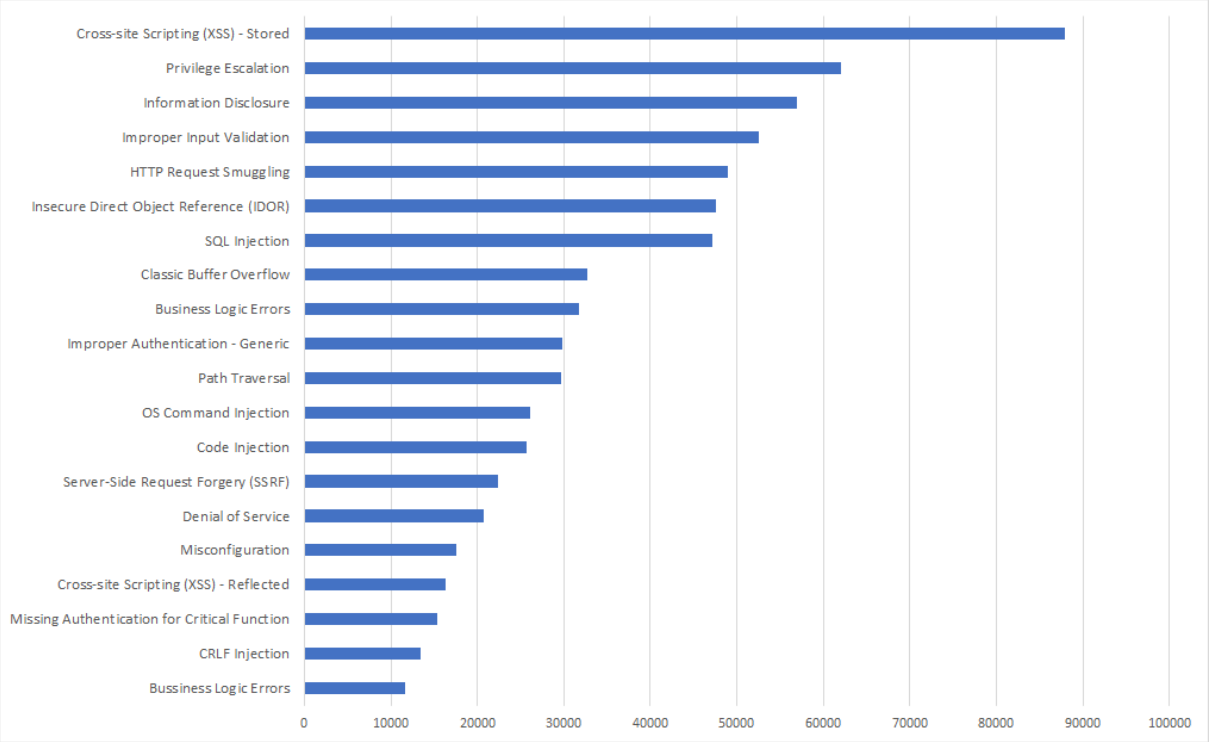


Fonte: ELABORADO PELO AUTOR

5.3 ANÁLISE DE BOUNTY POR VULNERABILIDADE

Na Figura 12 foi efetuado um levantamento levando em consideração o valor da recompensa paga por todas as empresas para cada tipo de vulnerabilidade. Nota-se que os top 5 apresentados (*Cross-site Scripting (XSS) Stored, Privilege Escalation, Information Disclosure, Improper Input Validation, HTTP Request Smuggling*) são os bugs mais bem pagos.

Figura 12: Soma Bounty x Vulnerabilidades

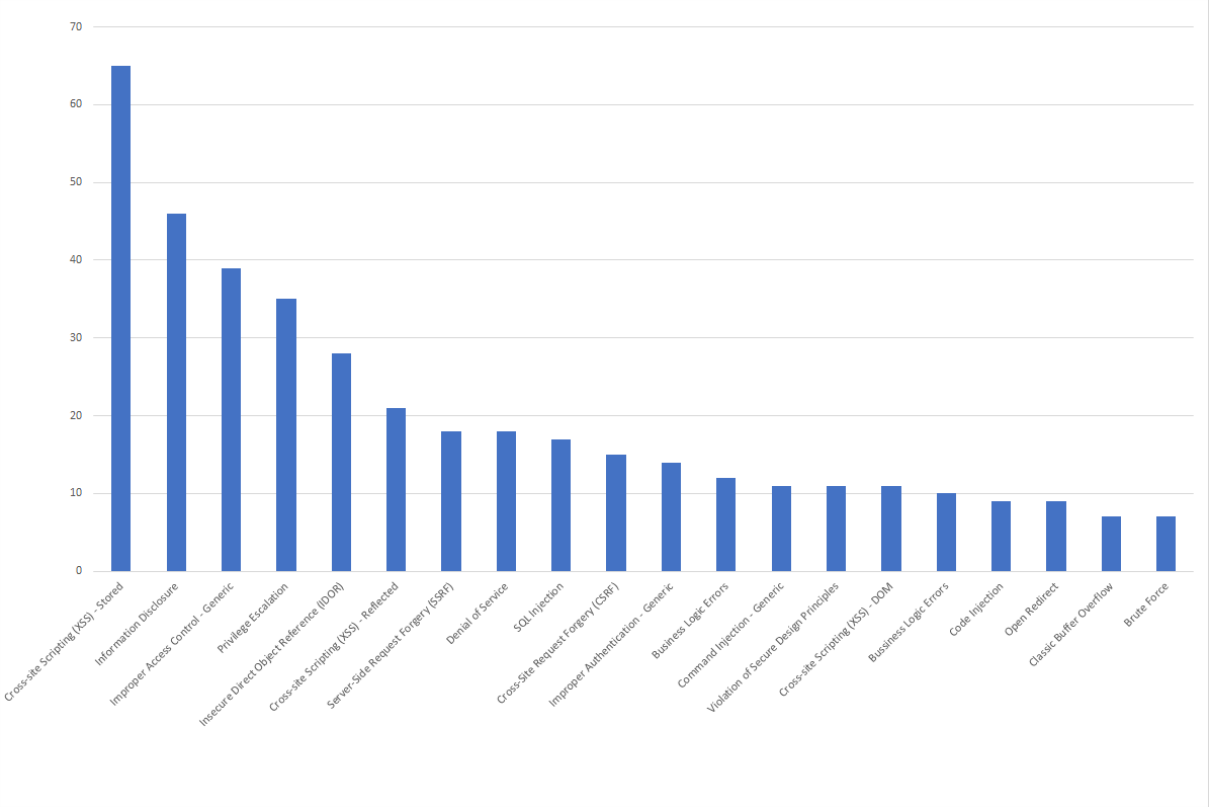


Fonte: ELABORADO PELO AUTOR

5.4 ANÁLISE DAS PRINCIPAIS VULNERABILIDADES

Já na Figura 13 as vulnerabilidades são demonstradas levando em consideração a quantidade de bugs gerados em todas as empresas. É possível notar que vulnerabilidades de XSS entram no top 5, pelo motivo dos sites de *ecommerce* estarem mais propensos a invasões, conforme visto na Figura 10.

Figura 13: Total Vulnerabilidades



Fonte: ELABORADO PELO AUTOR

6 CONCLUSÃO

Atualmente a grande maioria das empresas não utilizam os serviços de recompensas, os quais poderiam agilizar na correção de vulnerabilidades e para um desenvolvimento melhor de suas aplicações internamente e com isso diminuir o risco de invasões indesejadas e ocasionando a perda ou vazamento de dados.

Dentre os tipos de programas de recompensa, existem desde aqueles que possibilitam o retorno monetário como forma de pagamento, entre outros que podem ser reconhecimento ou bônus. Para que seja adotado o nível de serviço mais adequado, é preciso avaliar se a empresa pretende disponibilizar a exploração de todos os seus serviços disponibilizados ou somente incluir algumas aplicações ou serviços *webs*.

Cada um destes objetivos foram apresentados e investigados neste trabalho. Desta forma o público interessado em ingressar no BBP, tem informações sobre a área e quais vulnerabilidades focar primeiramente, visto que estão em destaque, são as que mais ocorrem nas grande maioria das empresas.

Além dos objetivos propostos também foram indicadas as principais vulnerabilidades que ocorreram no período analisado nos anos de 2019 e 2020, presentes nas plataformas como forma de direcionar possíveis esforços de analistas de segurança e testadores em posteriores buscas por novas vulnerabilidades. Também foi realizado o cruzamento de informações relacionadas as vulnerabilidades registradas nas plataformas de *Bug Bounty* selecionadas, como as empresas e os valores pagos pelas mesmas.

Conclui-se que um dos principais benefícios de utilizar o programa de recompensa é não depender somente do time interno, quanto mais pessoas estão analisando o ambiente, mais rapidamente suas vulnerabilidades são exploradas, com isso a empresa pode se dedicar mais ao desenvolvimento e correção.

Desta forma, os serviços não se tornariam vulneráveis com o passar do tempo e a empresa não terá a necessidade de grande mão de obra interna necessária para explorar vulnerabilidades, evitando o vazamento de informações restritas.

Afim da realização de trabalhos futuros, de maneira que possa seguir o mesmo foco ou complementar o trabalho proposto, pode ser realizado além da analise e demonstração das vulnerabilidades atuais, é realizar um estudo de vulnerabilidades e aplicá-las em empresas que tiverem interesse ou demonstrar a instituições a exploração de vulnerabilidades, também se estuda a possibilidade da criação de uma plataforma online, assim a contribuição com o público interessado é realizada com maior praticidade, os usuários podem acompanhar o que está em alta e o que já foi explorado.

REFERÊNCIAS

ABES. *Kaspersky Lab amplia programa de recompensa Bug Bounty*. 2018. Disponível em: <http://www.abessoftware.com.br/noticias/kaspersky-lab-amplia-programa-de-recompensa-bug-bounty>. Citado na página 16.

APPLE. *Apple Security Bounty*. 2020. Disponível em: <https://developer.apple.com/security-bounty/>. Citado 2 vezes nas páginas 26 e 31.

BARBOSA, D. C. *TOP 5 das vulnerabilidades mais frequentes em 2018*. 2019. Disponível em: <https://www.welivesecurity.com/br/2019/02/15/top-5-das-vulnerabilidades-mais-frequentes-em-2018/>. Citado na página 18.

BEKERMAN, D. e. S. Y. *The State of Vulnerabilities in 2019*. 2020. Disponível em: <https://www.imperva.com/blog/the-state-of-vulnerabilities-in-2019/>. Citado 2 vezes nas páginas 30 e 31.

BIENZ, C. e. S. J. *Software Vulnerabilities and Bug Bounty Programs*. 2020. Disponível em: <https://ieeexplore.ieee.org/abstract/document/9034828/authorsauthors>. Citado 2 vezes nas páginas 23 e 24.

BOHME, R. *Vulnerability Markets - What is the economic value of a zero-day exploit?* 2005. Disponível em: <http://events.ccc.de/congress/2005/fahrplan/attachments/542-Boehme2005_22C3_VulnerabilityMarkets.pdf>. Acesso em: 26 março 2019. Citado na página 8.

BUGCROWD. *Public Bug Bounty List*. 2020. Disponível em: <https://www.bugcrowd.com-/bug-bounty-list/>. Citado na página 26.

CANFIELD, C. e. F. C. e. N. R. 2015. Disponível em: <https://www.andrew.cmu.edu/user-/ccanfiel/National-Cyber-Bug-Broker_final.pdf>. Citado na página 22.

CHARLIE, M. *The legitimate vulnerability market: the secretive world of 0-day exploit sales. In 6th Workshop on the Economics of Information Security (WEIS 2007)*. 2007. Disponível em: <http://weis2007.econinfosec.org/papers/29.pdf>. Citado na página 10.

FACEBOOK. *Updates to the Bug Bounty Program - Why a Bug Bounty Program? Facebook Security*. 2011. Disponível em: <https://www.facebook.com/notes/facebooksecurity/updates-to-the-bug-bounty-program/10150270651335766>. Citado na página 15.

GOOGLE. *Google Application Security*. 2020. Disponível em: <https://www.google.com-/about/appsecurity/research/>. Citado na página 26.

HACKERONE. *Improving Public Bug Bounty Programs with Signal Requirements*. 2016. Disponível em: <https://www.hackerone.com/blog/signal-requirements>. Citado 3 vezes nas páginas 11, 12 e 13.

HACKERONE. *Bug Bounty Programs*. 2020. Disponível em: <https://hackerone.com/bug-bounty-programs>. Citado na página 26.

HARÁN, J. M. *Netflix lança um programa de Bug Bounty e oferece recompensas de até US$ 15 mil*. 2018. Disponível em: <https://www.welivesecurity.com/br/2018/03/27/netflix-lanca-um-programa-de-bug-bounty/>. Citado na página 15.

HATA, H. e. M. G. e. M. A. B. *Understanding the Heterogeneity of Contributors in Bug Bounty Programs*. 2017. Disponível em: <https://dl.acm.org/citation.cfm?id=3200529>. Citado 2 vezes nas páginas 22 e 24.

HUANG, K. e. M. S. e. S. M. e. X. L. e. Z. F. *Diversity or Concentration? Hackers' Strategy for Working Across Multiple Bug Bounty Programs*. 2016. Disponível em: <https://ieeexplore-.ieee.org/abstract/document/9034828/authorsauthors>. Citado 2 vezes nas páginas 23 e 24.

KASPERSKY. *Kaspersky*. 2020. Disponível em: <https://hackerone.com/kaspersky>. Citado na página 16.

KUEHN, A. e. M. M. *Analyzing Bug Bounty Programs: An Institutional Perspective on the Economics of Software Vulnerabilities*. [s.n.], 2014. Disponível em: <https://papers.ssrn.com-/sol3/papers.cfm?abstract_id=2418812>. Citado 7 vezes nas páginas 8, 10, 13, 14, 22, 23 e 24.

LASZKA, A. e. M. Z. e. J. G. *Banishing Misaligned Incentives for Validating Reports in Bug-Bounty Platforms*. 2016. Disponível em: <https://link.springer.com/chapter/10.1007/978-3-319-45741-3_9>. Citado 2 vezes nas páginas 10 e 11.

MAILLART, T. e. M. Z. e. J. G. e. J. C. *Given Enough Eyeballs, All Bugs are Shallow? Revisiting Eric Raymond with Bug Bounty Programs*. 2017. Disponível em: <https:/-/academic.oup.com/cybersecurity/article/3/2/81/4524054>. Citado 2 vezes nas páginas 13 e 14.

MICROSOFT. *Bounty Evolution: 100,000 for New Mitigation Bypass Techniques Wanted Dead or Alive. BlueHat Blog*. 2013a. Disponível em: <http://blogs.technet.com/b/bluehat-/archive/2013/11/01/bounty-evolution-100-000-for-newmitigation-bypass-techniques-wanted-dead-or-alive.aspx>. Citado na página 15.

MICROSOFT. *Heart of Blue Gold – Announcing New Bounty Programs. BlueHat Blog*. 2013b. Disponível em: <https://blogs.technet.microsoft.com/bluehat/2013/06/19/heart-of-blue-gold-announcing-new-bounty-programs/>. Citado na página 15.

MOHAN, D. S. M. V. *User Centric Security Models for Improving the Data Security from SQL Injections and Cross Site Scripting Attacks*. 2019. Disponível em: <http:/-/gujaratresearchsociety.in/index.php/JGRS/article/view/197/171>. Citado na página 19.

OWASP. *OWASP Top Ten*. 2020. Disponível em: <https://owasp.org/www-project-top-ten/>. Citado 4 vezes nas páginas 9, 18, 19 e 20.

RAMíREZ, F. *En auge los programas de 'Bug Bounty'*. 2020. Disponível em: <https:/-/unaaldia.hispasec.com/2020/07/en-auge-los-programas-de-bug-bounty.html>. Citado na página 14.

ROBERTSON, J. *Facebook Widens "Bug Bounty" Program to Combat Internal Breaches. Bloomberg*. 2012. Disponível em: <https://www.bloomberg.com/news/articles/2012-07-26-/facebook-widens-bug-bounty-program-to-combat-internal-breaches>. Citado na página 15.

SANTANA, B. *Apple fornecerá iPhones desbloqueados para pesquisa de segurança; programa de recompensas é expandido*. 2019. Disponível em: <https://macmagazine.uol-.com.br/post/2019/08/08/apple-fornecera-iphones-desbloqueados-para-pesquisa-de-seguranca-programa-de-recompensas-e-expandido/>. Citado na página 17.

SHULZ, P. *Penetration Testing of Web Applications in a Bug Bounty Program*. 2014. Citado na página 17.

SPRING, T. *HackerOne Offers Open Source Projects Free Access to Platform*. 2017. Disponível em: <https://threatpost.com/hackerone-offers-open-source-projects-free-access-to-platform/124070/>. Citado 2 vezes nas páginas 15 e 16.

TIINSIDE. *Kaspersky Lab lança programa de descobertas de bugs em parceria com HackerOne*. 2016. Disponível em: <https://tiinside.com.br/02/08/2016/kaspersky-lab-lanca-programa-de-edscobertas-de-bugs-em-parceria-com-hackerone/>. Citado na página 16.

WALSHE, T. *An Empirical Study of Bug Bounty Programs*. 2020. Disponível em: <https:/-/ieeexplore.ieee.org/abstract/document/9034828/authorsauthors>. Citado 2 vezes nas páginas 23 e 24.