Bachelorarbeit am Institut für Informatik der Freien Universität Berlin,

Arbeitsgruppe Software Engineering

Bug-Bounty-Programme im Softwareentwicklungsprozess - Chancen und Risiken -

Robert Philipps

Matrikelnummer: 4677023
Mail: robert.philipps@fu-berlin.de

Betreuer: Prof. Dr. Lutz Prechelt
Zweitgutachter: Prof. Dr.-Ing. Volker Roth

Berlin, den 03. Mai 2017

Zusammenfassung

Das Konzept, unabhängige Sicherheitsforscher_innen für das Entdecken unbekannter Sicherheitslücken in eigenen Applikationen zu entlohnen, gibt es schon seit einiger Zeit, dennoch wurde dieses bisher nur von wenigen Firmen umgesetzt. Dies liegt wohl zum einen in dem erheblichen Eingriff in den Softwareentwicklungsprozess und zum anderen darin, dass derzeit kaum Publikationen oder Dokumentationen vorhanden sind, die den grundsätzlichen Rahmen eines Bug-Bounty-Programms skizzieren und ein allgemeines Modell ableiten. In dieser Arbeit wird ein solches auf Grundlage einer großflächigen Marktanalyse konzipiert und die notwendigen Bestandteile und Prozesse herausgearbeitet. Hierfür werden auch die beiden Gruppen von Akteure mit unterschiedlichen Motivationsprofilen untersucht und bewertet, die sich aus der Dynamik eines Bug-Bounty-Programms ergeben.
Als Fallbeispiel dient der vorherige Zustand und die anschließende Implementierung bei eBay Kleinanzeigen, einer großen Online-Plattform.

Abstract

The concept of paying external security researchers for finding unknown security bugs is not extremely new, nevertheless it is not applied by a lot of companies. On the one hand this could be, because of the significant interference with the software engineering process, on the other hand it could be based on the fact that there are up to no publications or documentations that show the principal framework of bug bounty programs and deduce a generic theoretical model. In this thesis, this framework and model will be constructed with the help of an extensive market analysis and the needed components and processed will be worked out. This includes the different needs of the two opposing groups of players, that come from the unique dynamics of a bug bounty program.
eBay Kleinanzeigen, a big online platform, is used as a case example to show the status before and the implementation of a bug bounty program.

Eidesstattliche Erklärung

Ich versichere hiermit an Eides statt, dass diese Arbeit von niemand anderem als meiner Person verfasst worden ist. Alle verwendeten Hilfsmittel wie Berichte, Bücher, Internetseiten oder Ähnliches sind im Literaturverzeichnis angegeben, Zitate aus fremden Arbeiten sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungskommission vorgelegt und auch nicht veröffentlicht.

Robert Philipps	Ort, Datum

Erklärung zum Interessenkonflikt

Der Autor ist seit Dezember 2015 Angestellter von eBay Inc. im eBay Kleinanzeigen-Team, zuerst als Werkstudent im Bereich Backend-Development und seit Oktober 2016 als Software-Engineer. In dem in dieser Arbeit referenzierten Vorfall bezüglich einer XSS-Lücke war er der beteiligte Softwareentwickler. Er ist Mitglied der eBay Kleinanzeigen-Security-Guild und war als Hauptverantwortlicher maßgeblich an der Implementierung des Bug-Bounty-Programms beteiligt.
Diese Arbeit entstand nicht in Kooperation mit eBay, der eBay Classifieds Group oder eBay Kleinanzeigen und spiegelt nur die persönlichen Ansichten und Arbeitsergebnisse des Autors wieder, nicht die von eBay oder seinen Tochtergesellschaften. Es wurden von eBay Kleinanzeigen lediglich einzelne Inhalte für die Nutzung in dieser Abschlussarbeit freigegeben.

Robert Philipps	Ort, Datum
Rami Bader (GF eBay Kleinanzeigen GmbH)	Ort, Datum

Inhaltsverzeichnis

Abbildungsverzeichnis

In dieser Arbeit wurden einige Screenshots geschwärzt, da diese personenbezogene Daten oder sensible Unternehmensdaten enthielten.

Tabellenverzeichnis

1 Einleitung

> ***"Ask yourself the question: will there still be robbers even if you offer them jobs as lock pickers?"***
>
> *@Hackdwerg, ein White Hat Hacker aus den Niederlanden* [1]

Das Konzept, unabhängige Sicherheitsforscher[1] für das Entdecken von unbekannten Sicherheitslücken in eigenen Applikationen zu entlohnen, gibt es schon seit einiger Zeit. Bereits 1995 wurde von der Netscape Communications Corporation im Kontext ihres Netscape Navigator das erste Bug-Bounty-Programm eingeführt [2]. Ziel eines solches Programmes ist das standardisierte Verifizieren der Softwarelücke, ihre schnellstmögliche Behebung und das Entlohnen der externen Personen.

1.1 Problemstellung

Heute findet man aber meist nur in einigen der größten Softwareentwicklungskonzerne einen solchen klar definierten Prozess, der es konzernfremden Personen ermöglicht, Sicherheitslücken zu melden und der gleichzeitig internes Personal bei der Integration in den Softwareentwicklungsprozess unterstützt.
Betrachtet man die Unternehmen, die diesen Prozess nach außen kommunizieren und für die Öffentlichkeit klar darstellen, so reduziert sich die Anzahl nochmal erheblich. Das erscheint auf den ersten Blick verwunderlich, da Konzerne auf diese Weise das Risiko eingehen, über Sicherheitslücken nicht frühzeitig informiert zu werden um sie anschließend zügig beheben zu können, gerade weil oberflächlich betrachtet das Eröffnen eines Bug-Bounty-Programms als triviale Aufgabe erscheinen mag.
Tatsächlich bedeutet dies aber die Implementierung diverser Bestandteile und Prozesse, angefangen von der Meldung einer Sicherheitslücke, über die Entwicklung eines Software-Patches, hin zur möglichen Entlohnung. Dadurch ergibt sich ein nicht unerheblicher Eingriff in den Softwareentwicklungsprozess. Aufgrund der heterogenen Anforderungen können die einzelnen Prozesse zudem meist nur von verschiedene Personengruppen durchgeführt werden. Erfahrungen diverser Firmen zeigen, dass gerade die unsachgemäße Implementierung von Bug-Bounty-Programmen zu fatalen Folgen sowohl für die Sicherheit der Applikation als auch für den Softwareentwicklungsprozess führen kann.
Die Abwesenheit von Bug-Bounty-Programmen oder ihre unausgereifte Implementierung, lassen sich jedoch auch damit begründen, dass derzeit kaum

[1] Im Folgenden wird aus Gründen der besseren Lesbarkeit ausschließlich die männliche Form für jegliche Personengruppen benutzt. Dies dient nur der besseren Lesbarkeit und soll keine Einschränkung bedeuten. Es sind Menschen aller Geschlechter gemeint.

Publikationen oder Dokumentationen vorhanden sind, die den grundsätzlichen Rahmen eines Bug-Bounty-Programms skizzieren und eine praktische Implementierung in einen vorhandenen Softwareentwicklungsprozess dokumentieren. Die Bachelorarbeit soll dazu beitragen, diese Lücke zu schließen.

1.2 eBay Kleinanzeigen

Die anwendungsbezogenen Teile dieser Abschlussarbeit beschäftigen sich mit dem Bug-Bounty-Programm des Unternehmens eBay Kleinanzeigen.
eBay Kleinanzeigen ist eine Online-Plattform für Kleinanzeigen, ausgerichtet auf den deutschen Markt. Nach internen Unternehmensangaben hat es mit über 18 Millionen Nutzern pro Monat eine der zehn am häufigsten besuchten Websites Deutschlands [3] und mit über 20 Millionen aktiven Anzeigen ist es der größte Online-Kleinanzeigenmarkt Deutschlands.
eBay Kleinanzeigen ist als Teil der eBay Classifieds Group (eCG) eine hundertprozentige Tochterfirma von eBay Inc. und verfügt mehrheitlich über vollkommen autonome Abteilungen. So werden etwa die meisten Produkte, wie zum Beispiel die Website, die Apps oder auch die internen Kundensupport-Tools, eigenständig entwickelt. Dennoch besteht reger Wissenstransfer sowie Austausch jeglicher Ressourcen zwischen den einzelnen Produktabteilungen im Großkonzern eBay und besonders, durch die Nähe der Produkte, in der eBay Classifieds Group. Es werden auch diverse technische Produkte wie zum Beispiel Cloud-Infrastrukturen und Deployment-Systeme gemeinsam genutzt und gepflegt.

1.3 Vorgehensweise

Im nächsten Kapitel wird der Softwareentwicklungsprozess von eBay Kleinanzeigen vor Implementierung des Bug-Bounty-Programms anhand eines Vorfalls beschrieben und der Umgang mit Bug-Meldungen evaluiert.
Im Anschluss daran werden zentrale Begriffe erläutert sowie relevante Akteure vorgestellt, die als Basis für eine theoretische Betrachtung benötigt werden. Im Folgekapitel werden die einzelnen Bestandteile eines Bug-Bounty-Programms herausgearbeitet und die nötigen personellen sowie strukturellen Anpassungen für den Softwareentwicklungsprozess analysiert. Um die Attraktivität eines Bug-Bounty-Programms einschätzen zu können, folgt eine Analyse der Motivation der Akteure und Determinanten.
Daran anschließend wird unter Berücksichtigung der zuvor gewonnen theoretischen Erkenntnisse erläutert, wie das Bug-Bounty-Programm in den Softwareentwicklungsprozess von eBay Kleinanzeigen integriert wurde.
Die abschließende Diskussion fasst zentrale Erkenntnisse der Arbeit zusammen, bringt sie in den Kontext der aktuellen Entwicklungen und legt mögliche Fragestellungen zur weitergehenden Betrachtung der Thematik dar.

2 Stand vor Implementierung

2.1 Softwareentwicklungsprozess

Der Softwareentwicklungsprozess bei eBay Kleinanzeigen ist nach Produktbereichen sortiert. So gibt es ein Team, das sich um die Entwicklung der Website (sowohl mobil, als auch für den Desktop) kümmert, und weitere Teams, die sich zum Beispiel um die Entwicklung der iOS-App oder der Android-App kümmern. Die Arbeit der einzelnen Teams basiert auf jeweils zwei Kanban-Boards.

2.1.1 Kanban-Boards

Einerseits gibt es ein sogenanntes "Epic Board", auf dem größere Aufgabenkomplexe platziert werden, andererseits ein "Product Board", auf dem kleinere Aufgaben platziert werden. Diese sind entweder eigenständig oder Teile von Epics. Ein Ticket durchläuft mehrere Stufen vom (nicht dargestellten) Backlog bis hin zum Deployment im Produktivsystem.

Im Folgenden werden die einzelnen Spalten eines dieser Product-Boards (siehe Abbildung 1) kurz beschrieben:

ToDo
: Alle Tickets, die derzeit bearbeitet werden sollen, befinden sich in dieser Spalte. Die Tickets wurden, in einem für diese Abschlussarbeit nicht relevanten Prozess, von einem Team aus verschiedenen Abteilungen aus dem Backlog ausgewählt. Die Tickets sind priorisiert und werden je nach Kompetenzen der Entwickler von oben nach unten abgearbeitet.

Dev Ongoing
: In dieser Spalte befinden sich alle Tickets, an denen gerade ein oder mehrere Entwickler, zum Beispiel im Pair-Programming, arbeiten. Da die Entwicklung zur Versionsverwaltung Git in einer GitHub Enterprise Version verwendet, wird gewöhnlich auf einem Feature-Branch gearbeitet.

To Review
: Stellt ein Entwickler die Aufgabe seines Tickets fertig, wird ein Pull-Request kreiert und das Ticket von ihm auf "To Review" verschoben.

In Review
: Pull-Requests können von beliebig vielen Entwicklern kommentiert werden, müssen aber mindestens von einer Person offiziell bearbeitet werden. Sie schiebt das Ticket dann in diese Spalte und schlägt folgend gegebenenfalls Anpassungen vor oder approved den Pull-Request.

Abbildung 1: Übersicht des “Plattform”-Kanban-Boards

Ready to Merge
Wird der Pull-Request approved, landet das Ticket automatisch in der "Ready To Merge" Spalte. Diese wird konstant nach dem FIFO abgearbeitet.

Dev Done
Wurde ein Feature-Branch in den Master gemerged, befindet es sich in dieser Spalte. Während das Ticket hier liegt, wird das Deployment komplett automatisiert auf die verschiedenen Testumgebungen vorgenommen. Dabei wird der gesamte Code gebaut und auch diverse Unit-, Integrations-, Codestyle, sowie UI-Tests ausgeführt.

Ready for QA
Laufen alle eben genannten Tests durch, wird das Ticket automatisch in diese Spalte geschoben und verbleibt dort, bis ein Entwickler die Qualitätssicherung (QA) übernimmt.

QA Ongoing
Übernimmt ein Entwickler die QA, so schiebt er das Ticket in diese Spalte. Im Anschluss überprüft er dann die, vom Ticketersteller und dem Entwickler definierten QA-Kriterien. Im Zuge dessen wird die neu entwickelte Funktion getestet, sowie andere Funktionalitäten überprüft, die von der Änderung ungewollt beeinflusst sein könnten. Ist letzteres der Fall, wird die Änderung aus der Versionierung entfernt und das Ticket erneut auf "Dev Ongoing" geschoben.

QA Done
Ergibt sich im QA-Prozess kein Problem, wird das Ticket auf "QA Done" geschoben und von dort aus automatisch in alle Produktivsysteme deployed.

Im Prozess, der auf dem Kanban-Board basiert gibt es an keiner Stelle feste Rollen oder Teams, die grundsätzlich Aufgaben eines speziellen Typs übernehmen (müssen). Jede Spalte auf dem Board hat ein niedrig angesetztes Limit, wie viele Tickets sich maximal dort befinden dürfen. Damit kann gewährleistet werden, dass alle Aufgaben bearbeitet werden und es einen konstanten Fluss durch das Board gibt. So ist es zum Beispiel üblich, dass ein Entwickler ein Ticket auf "Ready to Merge" hat und es weiter voranbringen möchte, aber ihn gleichzeitig ein Ticket auf "Ready to QA" blockiert. Übernimmt er nun die QA dieses Tickets, kann er damit die Blockierung aufheben, sodass sein Ticket weiter durch den Prozess geleitet werden kann. Es gibt auch Limits, die andere erstrebenswerte Eigenschaften eines Softwareentwicklungsprozesses fördern sollen. So bewirkt etwa das Limit der "Dev Ongoing" Spalte, dass Entwickler ohne Aufgaben an diesem Punkt nach Kooperation suchen, zum Beispiel in Form von Pair-Programming oder Arbeitsteilung, da sie keine neues Ticket aus der "ToDo"-Spalte nehmen können.

2.1.2 Zero-Security-Bug-Policy

Bei eBay Kleinanzeigen herrscht eine sogenannte “zero security bug policy”, also eine Regelung, die besagt, dass zu keiner Zeit (bekannte) Sicherheitslücken vorhanden sein dürfen. Ist diese Maxime verletzt, also ein sicherheitsrelevanter Bug bekannt, rücken die dazu erstellten Tickets umgehend im Kanban-Board auf die höchste Prioritätsstufe. So kann gewährleistet werden, dass Sicherheitslücken schnellstmöglich behandelt werden. Dies gilt auch, wenn die “ToDo”-Spalte auf dem Board bereits das eigentliche Limit erreicht hat. Um Irritationen vorzubeugen werden keine Tickets entfernt, sondern ein Überladen als Sondersituation akzeptiert. Um darüber hinaus Aufmerksamkeit auf diese Aktion zu lenken, wird das beteiligte Team neben der sichtbaren Veränderung auf dem Kanban Board zusätzlich über den benutzten Instant-Messaging-Dienst informiert.

2.2 Security-Struktur

Die Software-Security im gesamten Konzern ist über spezielle Teams organisiert. Diese werden als InfoSec, Kurzform für “information scecurity”, bezeichnet. Ihre jeweiligen Verantwortungen werden hierarchisch über die Teams verteilt. So gibt es ein eBay-InfoSec-Team, diesem untergordnet ein eCG-InfoSec-Team und darunter für die jeweiligen Produkte entweder ein eigenständiges Team oder einen Mitarbeiter. Im Gegensatz zu Mitarbeitern, die in InfoSec-Teams arbeiten und keine anderen beruflichen Aufgabenfelder bearbeiten, tragen Mitarbeiter, die nicht in ein solches Team integriert sind, diese Verantwortung zusätzlich zu ihrer normalen Tätigkeit.
Historisch gewachsen gehören diese Teams bei eBay zu den SiteOps, also den Abteilungen, die für die tägliche Erreichbarkeit und Funktionalität der Website, sowie für die Infrastruktur verantwortlich sind. Da die meisten Security-Themen jedoch oft keinerlei Bezug zu den Aufgaben eines SiteOps-Ingenieurs haben, sondern thematisch im Feld der klassischen Softwareentwicklung zu verorten sind, wuchs der Bedarf nach einem zusätzlichen Security-Team. Aus diesem Grund haben einige eBay-Produkte eine Guild[2] zum Thema Security ins Leben gerufen. Bei eBay Kleinanzeigen besteht diese Guild aus Mitarbeitern diverser technischer Abteilungen. Die Kommunikation erfolgt über einen speziellen Bereich im Instant-Messaging-Dienst, einen E-Mail-Verteiler, einen Bereich im Wiki sowie über ein reguläres wöchentliches Meeting. Sofern erhöhter Aufwand besteht, werden Aufgaben an einen oder mehrere Mitglieder distribuiert und die ganze Guild anschließend über die Arbeitsergebnisse informiert.

[2] eBay Kleinanzeigen verwendet das von Spotify entwickelte Guild-Konzept. Dort gelten für eine Guild folgende Rahmenbedingungen: Freiwillige Mitgliedschaft, keine Restriktion bezüglich der Abteilung, Rolle, Hierarchiestufe oder Kompetenzen und selbst organisierend [4].

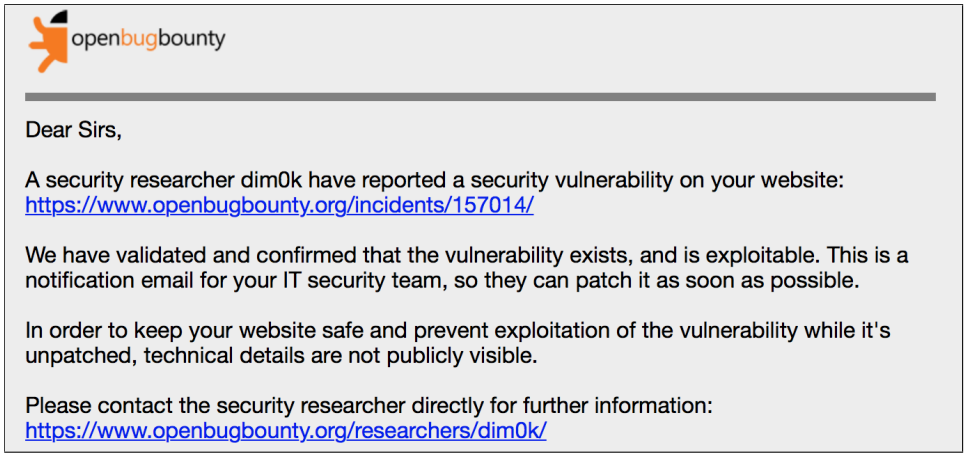


Abbildung 2: Ausschnitt der ersten Mail an den Kundensupport (Für die gesamte Mail siehe Anhang A.1)

2.3 Umgang mit Bug-Meldungen

Derzeit gibt es keinen geregelten Prozess für den Umgang mit Bug-Meldungen. Die Probleme, die sich daraus ergaben, lassen sich beispielhaft an einem Fall vom Mai 2016 illustrieren:
Ein Hacker fand eine Sicherheitslücke[3] in der eBay Kleinanzeigen Website, hatte aber keine Möglichkeit die Betreiber über diese zu informieren. Daher publizierte er das Vorhandensein der Sicherheitslücke auf openbugbounty.org, einer öffentlichen, unkommerziellen Plattform zur Aufnahme von Sicherheitslücken. Daraufhin sendete diese automatisiert standardisierte E-Mails an zahlreiche generische E-Mail-Konten der betreffenden Domain. Eine dieser E-Mails erreichte das System des Kundensupports unter info@ebay-kleinanzeigen.de (Siehe Abbildung 2).
Dort wurde die Mail vom First-Level-Support[4] weiter an den Second-Level-Support eskaliert.
Aufgrund der möglichen Brisanz wurde dann umgehend ein erreichbarer Softwareentwickler kontaktiert. Dieser bestätigte die Seriosität der Anfrage und übernahm die Bearbeitung. Umgehend wurde das Problem zu höheren Managementebenen bis zum eBay Kleinanzeigen-CTO und dem eCG-InfoSec-Verantwortlichen eskaliert, da von Seiten des Entwicklers Unsicherheit be-

[3] Es handelte sich um eine nicht persistente XSS-Injection, über die eBay Kleinanzeigen am 30.05.2016 gegen 18h informiert wurde. Knapp zwei Tage später, am 01.06.2016 um 10:41h, wurde der Fix zum Beheben der Vulnerability ins Produktivsystem eingespielt [Siehe Anhang B]. Die Logs wiesen keine Ausnutzung dieser Lücke aus. Es kann also davon ausgegangen werden, dass keine Nutzer betroffen waren.

[4] First-Level-Support ist die erste Anlaufstelle für eingehende Kundenanfragen. Können diese hier nicht abschließend bearbeitet, so werden sie an den Second-Level-Support eskaliert, der für komplexere Aufgaben zuständig ist [5].

züglich des weiteren Vorgehens herrschte. Daraufhin wurde der Hacker mit der von openbugbounty.org bereitgestellten E-Mail-Adresse kontaktiert. In seiner Antwort legte er die bestehende Sicherheitslücke offen (Für den gesamten Kommunikationsverlauf siehe Anhang A). Diese wurde umgehend validiert und es wurde nach einem Softwareentwickler gesucht, der gerade freie Kapazität hatte, sich des Problems anzunehmen. Nachdem ein Entwickler mit Expertise in dem Bereich sich bereit erklärte den Fall zu übernehmen, konnte die Sicherheitslücke zeitnah behoben werden.
Schon in seiner ersten Antwort fragte der Hacker nach einem Vorhandensein eines Bug-Bounty-Programms innerhalb eBay Kleinanzeigens, welches verneint werden musste. Unter den beteiligten Personen herrschte Konsens, dass man den Hacker für sein Tun monetär gerne entlohnen würde. Dies stellte aber ein unüberwindbares Hindernis innerhalb des Rechnungsablaufs von eBay Kleinanzeigen respektive eBay dar. Als Alternative wurde entschieden, dem Hacker ein Empfehlungsschreiben von eBay auszustellen und ein Merchandise-Paket zuzusenden. Auch letzteres stellte eine große Herausforderung dar. Für den russischen Zoll waren einerseits vorab Einfuhrabgaben auf den Warenwert zu entrichten, anderseits musste für den Zoll jeder Artikel aufgelistet und preislich gekennzeichnet werden. Dies war bei dem großen Fundus an vorhandenem Merchandise mit zum Teil unbekannter Herkunft eine zusätzliche Erschwernis. Nach gut sieben Wochen erreichte das Paket den Hacker.

2.4 Evaluation des Umgangs

Der oben beschriebene Ablauf weist keinerlei Struktur auf und hatte daher zahlreiche unregulierte Schnittstellen, an denen die Gefahr von Informationsverlust bestand. So hätte die Anfrage des Hackers, beziehungsweise von openbugbounty.org, schon zu Beginn ins Leere laufen können, da ohne eine klare Kommunikationsschnittstelle generische E-Mail-Adressen zur Kontaktaufnahme genutzt wurden. Dies birgt zum einen das Risiko, dass keine der verwendeten E-Mail-Konten tatsächlich in Verwendung sind. Zum anderen besteht die Möglichkeit, dass der First-Level-Support des Kundendienstes die Anfrage fälschlicherweise als Spam einordnet und damit jede weitere Bearbeitung verhindert. Aufgrund des hohen täglichen Umfangs an Nachrichten können jeder Kundenanfrage nur begrenzte Ressourcen zur Verfügung gestellt werden. Selbiges gilt vermindert nach der Eskalation auch für den Second-Level-Support.
Im nächsten Schritt war nicht geregelt, wie der Kundensupport die Entwickler für Sicherheitsfragen kontaktiert. Glücklicherweise hatte ein Entwickler sofort Zeit, sich der Lücke anzunehmen und war auch im Verlauf bis zum letzten Kontakt zum Hacker erreichbar. Ein Ausfall, zum Beispiel durch Krankheit oder Urlaub, wäre kritisch gewesen, da keine für andere Mitarbeiter einsehbare Dokumentation existierte.

Nachdem ein Entwickler gefunden war die Anfrage zu übernehmen und die Legitimität der Meldung bestätigte, fehlte ein Vorgang, der festlegte, wie mit dem Hacker zu kommunizieren sei und welche Personen zu informieren wären. Daher musste einerseits erst eine Strategie für die Kommunikation mit dem Hacker entwickelt werden und anderseits mussten hohe Managementpositionen kontaktiert werden, um diesen neuartigen und unbekannten Prozess sicher begleiten zu lassen.
Da die Lücke von einem Softwareentwickler außerhalb des regulären Softwareentwicklungsprozesses behoben wurde, bestand dort eine ähnliche Problematik wie beim ersten Entwickler, der den Kontakt zum Hacker übernahm. Ebenso gab es an dieser Stelle keine Dokumentation und der Vorgang war somit nicht gut nachvollziehbar für andere Teammitglieder.
Bezüglich der Honorierung des Hackers konnte eBay Kleinanzeigen nicht wie geplant eine Belohnung auszahlen, da dies eine zu große unternehmensrechnerische Hürde darstellte, sondern musste auf das Zuschicken von Merchandise-Artikeln und ein Endorsement auf openbugbounty.org (Siehe Anhang D)für den Hacker zurückgreifen. Ersteres stellte sich hierbei als zeitaufwändiges Vorgang dar, da keinerlei Information über die Versandbesonderheiten ins Ausland bei den beteiligten Personen vorhanden war.
Im Nachhinein lässt sich sagen, dass die zeitnahe Behebung der Lücke und die gute Zusammenarbeit mit dem Hacker nicht durch einen geregelten Prozess hervorgerufen worden sind, sondern dessen Fehlen nur durch die durchlässigen und flachen Kommunikationsstrukturen und das ausgeprägte Prozessverständnis und Adaptionsvermögen der einzelnen Mitarbeiter kompensiert werden konnte. Ein solch positiver Ausgang in Abwesenheit eines geregelten Prozesses lässt sich jedoch keinesfalls bei jeder Bug-Meldung und vor allem nicht bei erhöhtem Aufkommen von Meldungen garantieren. Es ist im Gegenteil von einem hohen Risiko auszugehen, den Softwareentwicklungsprozess zu schädigen, und einen nicht zufriedenstellenden Umgang mit von Hackern übermittelten Sicherheitslücken zu haben.

Daher entschied sich das Management-Team von eBay Kleinanzeigen dazu, sich mit der Problematik von Bug-Meldungen näher auseinander zu setzten und dahingehend eine bessere Bewältigungsstrategie zu entwickeln. Dazu sollten die Wirkmechanismen, Prozesse und Bestandteile von Bug-Bounty-Programmen analysiert werden. Im folgenden Kapitel werden die dafür notwendigen Grundlagen erläutert.

3 Grundlagen

3.1 Defintionen

Zhao und Kollegen gehen in ihrer wirtschaftlichen Nutzenkalkulation bezüglich Bug-Bounty-Programmen [6] davon aus, dass sich in allen Softwareprodukten mit sehr hoher Wahrscheinlichkeit unentdeckte Bugs befinden. Um Diese zu beheben oder ihre Auslieferung in ein Produktivsystem zu verhindern, gibt es nach dem von Austin beschrieben Softwareentwicklungsprozess "secure software development lifecycle" [7] vier verschiedene Möglichkeiten:

Code-Reviews, bei denen ein oder mehrere Softwareentwickler den entstandenen Programmcode analysieren und gegebenenfalls freigeben

Penetration-Testing, bei dem internes oder externes Personal versucht von außen, ohne Einblick in den Programmcode, Bugs aufzudecken

Dynamische und statische Analysetools, die Programmcode oder eine gestartete Applikation auf verbreitete Schwachstellen untersuchen

Vulnerability-Rewards-Programme

Letztere, im Folgenden Bug-Bounty-Programme genannt [5], sind nach Tuttle Programme, in denen Unternehmen Hackern monetäre Anreize sowie öffentliche Anerkennung für das Melden von Schwachstellen im Softwareprodukt zusprechen [8, S. 6].

3.2 Akteure

Aus der Dynamik dieser Interaktion ergeben sich primär zwei Arten von Akteuren innerhalb fertiggestellter Bug-Bounty-Programme. Einerseits gibt es Stakeholder des Softwareproduktes; dies können Führungskräfte, auf Softwaresicherheit spezialisierte Angestellte oder gemeine Softwareentwickler sein. Im Interesse dieser Gruppe liegt es, möglichst wenige Bugs in der eigenen Softwarelösung zu haben. Nur so kann die Sicherheit und Funktionalität der IT-Infrastruktur und Daten gewährleistet werden, sowohl für den internen Gebrauch als auch für deren Inanspruchnahme durch Nutzer.
Auf der anderen Seite gibt es Hacker[6], also unternehmensfremde Personen, die aktiv nach Sicherheitslücken suchen.

[5] Diese Bezeichnung bildet nicht die klare Konzentration der Programme auf sicherheitsrelevante Bugs ab. Es entspricht jedoch der geläufigen Bezeichnung sowohl in der Wirtschaft als auch in der Fachliteratur.

[6] Der Begriffs des Hackers ist nicht klar definiert und hat je nach Kontext in der Fachliteratur eine unterschiedliche Bedeutung. Im weiteren Verlauf dieser Arbeit wird damit eine Person beschrieben, die sich mit Hilfe ihrer informationstechnologischen Fähigkeiten Zugriff zu fremden Systemen beschafft und nicht, wie ebenso gebräuchlich, einen allgemein hochqualifizierten Computerexperten. [9]

4 Aufbau eines Bug-Bounty-Programms

Bug-Bounty-Programme haben von der initalen Meldung bis hin zur Behebung der Lücke und Belohnung des Hackers eine Vielzahl an Komponenten, die alle sorgfältig justiert werden müssen, um den Erfolg eines solchen Programms zu gewährleisten. Aus diesem Grund wurde eine Analyse von diversen Bug-Bounty-Programmen vorgenommen, in der diese hinsichtlich ihrer Eigenschaften und Prozesse untersucht wurden. Hierbei wurde festgestellt, dass sich die Programme zwar hinsichtlich ihrer konkreten Umsetzung unterscheiden, aber in der Regel, und damit auch unabhängig vom jeweiligen Softwareentwicklungsprozess, die gleichen Herausforderungen bewältigen müssen. Daraus wurde ein theoretisches Model abgeleitet, welches die Bestandteile und Prozessabläufe in einem solchen Programm abbildet und erläutert, welche Kompetenzen jeweils gefordert werden. Dieses theoretische Model wird im Folgenden vorgestellt.

4.1 Bestandteile

Aus der eben genannten Analyse ergeben sich folgende Bestandteile:

4.1.1 Kommunikation über Vorhandensein

Über einen öffentlich einsehbaren Bereich wird kommuniziert, dass ein solches Programm vorhanden ist und welchen Regeln und Rahmenbedingungen es unterliegt. Da Hacker sich oftmals unabhängig von der Interfacesprache des Produktes im digitalen Raum bewegen, wird diese Information trotz Regionalbezugs der Software meist (auch) in Englisch formuliert. Darüber ist sie über gängige Suchmaschinen mit der Suchanfrage “[NAME_DES_PRODUKTES] bug bounty” zu finden.
Die Rahmenbedingungen beinhalten meist einen klaren Anwendungsbereich, also spezielle Domains, Softwareanwendungen oder zu benutzende Drittsoftware wie Internetbrowser und ein klares inkludiertes Spektrum an Bug-Arten wie zum Beispiel “Code Execution”, “XSS” oder “Authentication Bypass”. Sofern das Programm Zahlungen als Belohnung vorsieht, wird häufig eine genaue Auflistung angegeben, wieviel für Sicherheitslücken welcher Art gezahlt wird, wie zum Beispiel bei Magento [10]. Ist dies nicht der Fall so wird zumindest der vage monetäre Rahmen skizziert, wie etwa bei MasterCard [11]. Außerdem wird Hackern oft erklärt, welche Vorgänge sie auf dem Zielsystem zur Überprüfung ausführen dürfen, welche Daten berührt werden dürfen und in welche Systeme sie keinesfalls eingreifen dürfen.

4.1.2 Schnittstelle für Kommunikation

Es gibt mindestens eine bidirektionale Schnittstelle zur Kommunikation mit Stakeholdern des Produkts. Üblicherweise wird hierfür ein spezieller Bereich eingeführt oder es wird mit einer speziell dafür aufgesetzten E-Mail-Adresse kommuniziert. Diese Schnittstelle sollte so konzipiert sein, dass Produkt-Stakeholder über neue Nachrichten informiert werden, damit eine schnelle Reaktion auf Hackeranfragen gewährleistet werden kann. Beispielhaft für das Ergebnis mangelhafter Kommunikation ist ein Vorfall von Google aus dem Jahr 2003. Nachdem das Unternehmen über 40 Tage nicht mehr auf die Meldung einer Sicherheitslücke im Android-Betriebssystem reagierte, entschieden sich ihre Entdecker, diese zu publizieren [12].
Aufgrund der Brisanz der Informationen wird meist angeboten, die Kommunikation mit gängigen Verschlüsselungs- und Signierungsverfahren abzusichern [13]. Alternativ gibt es Firmen wie Symantec oder Dell, welche die Kommunikation bezüglich Sicherheitslücken nur bei Verwendung dieser erlauben [14][15].

4.1.3 Bug-Entgegennahme

Wurde ein potentieller Bug von einem Hacker übermittelt, so wird im nächsten Schritt geprüft, ob es sich dabei tatsächlich um eine Sicherheitslücke handelt. Oftmals wird die Kontaktmöglichkeit eines Bug-Bounty-Programm von Nutzern fälschlicherweise auch verwendet, um sich hinsichtlich nicht sicherheitsrelevanter Bugs oder genereller Anfragen, wie etwa Produktverbesserungsvorschlägen, an das Unternehmen zu wenden. Um dies zu verhindern, haben Firmen, wie zum Beispiel Google, eine Reihe von Fragen zur möglichst eindeutigen Identifikation sicherheitsrelevanter Bugs in den Prozess eingebaut. Auf diese Weise können Anfragen, welche nicht in den Zuständigkeitsbereich eines Bug-Bounty-Programmes fallen, abgefangen und mit automatisierten Lösungsvorschlägen versorgt werden (siehe Anhang C). Trotz dieses detaillierten Verfahrens gibt Google die Rate der Meldungen, die keinen Bezug zu sicherheitsrelevanten Bugs haben, mit etwa 90% an [16]. Es ist folglich davon auszugehen, dass Bug-Bounty-Programme, die nicht über ein solches Filterungssystem verfügen, mit einem noch höheren Anteil an irrelevanten Meldungen zu kämpfen haben. Insbesondere in Hinblick auf die personellen Ressourcen, welche ein Unternehmen für die Prüfung dieser Anfragen abstellen muss, stellt die Identifikation sicherheitsrelevanter Bug daher ein zentrales Element in der wirtschaftlichen Implementierung eines Bug-Bounty-Programmes dar.
Handelt es sich aber um eine legitime Meldung, so wird überprüft, ob alle notwendigen Angaben gemacht wurden. Dies beinhaltet in Abhängigkeit der Bedürfnisse des jeweiligen Programms eine Klassifizierung des Bugs so-

wie eine Anleitung zum Reproduzieren des Bugs. Unter Umständen werden auch die Nennung potentieller Angriffsszenarien, Auswirkungen der Sicherheitslücke, oder sogar eine mögliche Lösung zur Behebung des Bugs verlangt. Werden die gestellten Anforderungen an die Inhalte einer Bug-Meldung nicht erfüllt, wird der Hacker dazu aufgefordert, diese zu vervollständigen. Viele Firmen, wie etwa Twitter oder PayPal, verfügen über eine solche Liste konkreter Anforderungen [17][18]. Google hat für sein Programm sogar eine eigene Lehrwebsite zu diesem Thema ins Leben gerufen, die "Google Bughunter University" [19].

4.1.4 Bug-Verifizierung

Nach der erfolgreichen Aufnahme des Bugs wird im nächsten Schritt verifiziert, ob sich der Bug im Anwendungsbereich befindet, also je nach Programm zum Beispiel nicht auf Testumgebungen oder in externen Softwarebibliotheken. Ist Letzteres der Fall, wird der Hacker über die Ablehnung informiert. Andernfalls wird an dieser Stelle überprüft, ob sich die Sicherheitslücke tatsächlich im System befindet und ausnutzbar ist.
Hierbei wird der gemeldete Bug auch mit bereits gemeldeten Bugs verglichen und gegebenenfalls als Duplikat abgewiesen. Dies kann gerade bei populären Bug-Bounty-Programmen Frustration auf beiden Seiten herbeiführen, da zwischen internem Bekanntwerden und dem Beseitigen der Lücke je nach Priorität einige Tage bis Wochen liegen können [20][21]. Sofern also auch andere Hacker in diesem Zeitintervall die Sicherheitslücke finden, wird diese unwissentlich erneut eingereicht.

4.1.5 Vorfallüberprüfung

Als nächstes wird, sofern technisch möglich, geprüft ob und in welchem Ausmaß diese Lücke bereits ausgenutzt wurde. Dies erfolgt bei serverbasierter Software meist durch die Auswertung von Logs. Bei Anwendungen, die auf Kundensystemen ausgeführt werden, stellt sich dieser Prozess in der Regel erheblich schwerer da, kann aber meist durch Anfragen oder Erweiterungen der Software vorgenommen werden. Ob bis zu diesem Zeitpunkt von einer Ausnutzung der Schwachstelle auszugehen ist, steht im Zentrum der folgenden Priorisierung und weiterer einzuleitender Schritte[7].

[7]Sofern die Sicherheitslücke in der Vergangenheit bereits ausgenutzt wurde oder aktuell ausgenutzt wird, werden je nach Art der Lücke drastische Schritte eingeleitet. Es werden etwa betroffene Nutzer informiert, Backups eingespielt, die Presse kontaktiert; im Extremfall erfolgt auch die Komplettabschaltung des Systems. Im Folgenden werden diese Maßnahmen, welche den Umgang mit Schäden, die vor Bekanntwerden der Sicherheitslücke entstanden sind, behandeln, aufgrund ihrer Vielfältigkeit und fallgebunden Komplexität nicht näher differenziert.

4.1.6 Softwareanpassung

Schließlich wird die Entwicklungsabteilung informiert, um die Anforderungen der Anpassung in den Ablauf des Softwareentwicklungsprozesses zu integrieren. Dies kann je nach Komplexität und verwendeten Prozessframeworks und Softwarelösungen, zum Beispiel bei Scrum die Integration in den nächsten Sprint, bei Kanban die Erstellung einer Aufgabe oder im Wasserfallmodell den Start einer neuen Kaskade bedeuten. Häufig werden Sicherheitslücken priorisiert vor jeglichen anderen Anforderungen behandelt, sodass das Softwaresystem im Idealfall immer ohne (bekannte) Sicherheitslücken läuft. Die Priorisierung kann im Extremfall zu einem Ausbruch aus dem eigentlich verwendeten Prozess führen, um die zeitnahe Behebung des Bugs sicherzustellen.
In einigen Fällen kommt es vor, dass durch das Korrigieren eines Bugs die beteiligten Entwickler Gemeinsamkeiten mit anderen Stellen in der eigenen Softwareapplikation erkennen und dadurch zusätzliche (nicht gemeldete) Sicherheitslücken aufgedeckt werden können.

4.1.7 Fix-Validierung

Ist der Bug laut Entwicklungsabteilung behoben, wird nun erneut mit dem Hacker Kontakt aufgenommen und dieser aufgefordert, das erfolgreiche Beheben der Sicherheitslücke zu bestätigen. Dieser Schritt ist meist obsolet, da er in in den meisten Softwareentwicklungsprozessen bereits durch diverse qualitätssichernde Vorgänge abgedeckt wird. Im Zweifelsfall können dadurch jedoch Verständnisprobleme vorheriger Schritten aufgedeckt werden. So kann es sein, dass der Hacker die Lücke nicht vollumfänglich kommuniziert oder Mitarbeiter des Bug-Bounty-Programms dies nicht korrekt übernommen haben. In diesem Fall beginnt der Prozess wieder bei der Verifizierung.

4.1.8 Belohnungs- und Publikationsfestsetzung

Ist die Sicherheitslücke nach Auffassung beider Seiten behoben, so muss spätestens hier vom Unternehmen festgelegt werden, ob und in welcher Form eine Belohnung erfolgen soll. Sofern das Unternehmen monetäre Entlohnungen vorsieht, basiert die Auszahlungssumme meist auf einer publizierten Matrix, die je nach Kategorisierung eine entsprechende Höhe vorsieht. Oftmals müssen für den Bezahlvorgang komplexe Prozesse gestartet werden, damit eine unternehmensinterne Abrechnung möglich ist. So ist für eine Firma aus rechtlichen Gründen eine (hohe) Zahlung meist nur in Kombination mit einer formell korrekten Rechnung durchführbar. Außerdem sind Konzerne je nach Firmensitz an politische Restriktionen, wie zum Beispiel Wirtschaftsembargos gebunden, und können nicht mit juristischen Personen öffentlichen Rechts jeglicher Nationalität Verträge eingehen [22].

Oft wird als zusätzliche Anerkennung zur monetären Entlohnung oder anstelle dieser Merchandise an den Hacker versandt. Entweder wird ein standardisiertes Set an Werbeartikeln zugeschickt oder dem Hacker die Möglichkeit gegeben, eine personalisierte Auswahl an Gegenständen kostenfrei über einen Online-Shop zu bestellen. Der Versand kann aber, selbst bei niedrigem Warenwert, eine große Herausforderung für den Konzern bedeuten. Je nach Standort des Hackers und des Unternehmens müssen spezielle Anforderungen für den Versand sowie für Einfuhrzölle erfüllt werden.
Je nach Bug-Bounty-Programm wird entweder auf Anfrage des Hackers oder regulär standardisiert entschieden, ob die Sicherheitslücke vom Unternehmen publiziert wird, beziehungsweise vom Hacker publiziert werden darf. Hier sind Zusammenfassungen oder zensierte Veröffentlichungen üblich. Bei diesen werden spezielle Informationen, die unter den umstrittenen Bereich "Security through obscurity" fallen würden oder aus geschäftlichen Gründen intern bleiben sollen, damit vor der Öffentlichkeit zurückgehalten.

4.2 Kompetenzmatrix

Die Aufgaben, die sich aus den verschiedenen Bestandteilen ergeben, erfordern verschiedene Kompetenzen und Zugriffsrechte. Im Folgenden werden die dafür erforderlichen Profile in vier unterschiedlichen Überbereichen zusammengefasst.

Personen mit Kommunikationskompentenz (PK), die geschult sind, mit außenstehenden Personen oder der Presse im Interesse der Firma über unternehmensrelevante Inhalte zu kommunizieren, zum Beispiel Mitarbeiter der Presse- oder Marketingabteilung.

Softwareentwickler (SWE), die grundsätzliches Verständnis für die Anwendungsdomäne der Applikation haben, aber nicht an der verwendeten Software arbeiten müssen. Sie können mit gängigen Tools umgehen und technische Beschreibungen verstehen und umsetzen.

SWE mit Zugang zum Produktionssystem (SWE+P), Entwickler mit Zugriff auf die Produktiv- und Testsysteme, um Logs einzusehen und Auswirkungen auf den verwendeten Umgebungen zu bewerten.

SWE mit Zugang zum Code(SWE+C), Entwickler, die an Teilen der Applikation arbeiten, in denen die Sicherheitslücke auftritt. Sie haben Zugriff auf den vollen Programmcode und die Kompetenz, diesen zu verstehen und gegebenenfalls zu ändern.

Die Unterteilung zwischen einem gemeinen SWE und einem SWE+P oder SWE+C wird vorgenommen, da je nach Größe einer Softwareentwicklungsabteilung, Kultur eines Konzerns oder Art der Applikation (zum Beispiel

statische Website oder personenbezogene Verschlüsselungssoftware) das Einsehen des Programmcodes oder der Zugang zu Produktivsystemen aus Sicherheitsgründen stark restriktiert wird. Man versucht damit, möglichst viele Aufgaben, für die kein Code- oder Produktivsystemzugriff benötigt wird, auf Mitarbeiter mit weniger Rechten auszulagern. Nur so können die SWE+Ps und SWE+Cs, als begrenzte Ressource, tatsächlich ausschließlich Aufgaben übernehmen, für die diese Rechte notwendig sind.
Betrachtet man nun, wie die Kompetenzverteilung der einzelnen und für jede Meldung wiederkehrenden Bestandteile eines Bug-Bounty-Programms aussehen könnte, kommt man auf die in Tabelle 1 dargestellte Matrix.

	PK	SWE	SWE+P	SWE+C
Bug-Entgegennahme	x			
Bug-Verifizierung		x		
Vorfallüberprüfung			x	
Softwareanpassung				x
Fix-Validierung	x			
Belohnungs- und Publikationsfestsetzung	x			

Tabelle 1: Kompetenzmatrix für Bug-Bounty-Programme.

Auf der einen Seite werden für die drei Kontaktpunkte mit den Hackern, also zuerst der Entgegenahme des Bugs, die spätere Validierung des Fixes und zuletzt die Festsetzung der Belohnung und der Publikation, andere Kompetenzen und Rechte benötigt, als für die übrigen Bestandteile. Dies lässt sich durch die homogenen Aufgabenbereiche der drei Bestandteile erklären, denn diese zeichnen sich dadurch aus, dass sie oftmals die einzige Informationsquelle für den Hacker oder gegebenenfalls auch die interessierte Öffentlichkeit sind. Daher ist der richtige Umgang und die sorgfältige Auswahl der zu publizierenden Informationen essentiell für den Ausgang eines Bug-Bounty-Programms.
Auf der anderen Seite gibt es die drei Bestandteile, die sich ausschließlich von Personal mit technischen Kompetenzen durchführen lassen. Im Gegensatz zur Anpassung der Software und der Überprüfung ob der Ausnutzung der Sicherheitslücke, wird für die Verifizierung eines Bugs kein Zugriff zu Produktivsystemen und keine Kenntnis des Programmcodes benötigt. Da hier nur von außen, wenn benötigt mit Hilfe von Test-Accounts und diversen Testgeräten geprüft wird, ob der Bug wie beschrieben ausnutzbar ist, sind nur geringe Rechte gefordert.
Wie der obenstehenden Tabelle zu entnehmen ist, ergibt sich aus den einzelnen Bestandteilen ein breites Spektrum benötigter Kompetenzen. Je nachdem, ob zwischen SWE, SWE+P und SWE+C getrennt wird, besteht die

Kompetenzmatrix aus zwei bis vier Bereichen. Alle Bestandteile können theoretisch von nur einer Person beziehungsweise Personen einer Rolle abgedeckt werden, sofern dort alle geforderten Kompetenzen gebündelt werden. Dies ist aber in der Realität vermutlich eher durch die Zusammenarbeit mehrerer Personen und Abteilungen im Unternehmen erreichbar.

4.3 Art der Implementierung

Um die Zusammenarbeit zwischen verschiedenen Rollen und der gegebenenfalls großen Anzahl beteiligter Personen zu ermöglichen, bedarf es der Implementierung von Kommunikations- und Aufgabensystemen sowie der Kreation diverser geregelter Prozesse. In Bezug auf die Umsetzung der Bestandteile der Bug-Bounty-Programme und ihrer Integration in den Softwareentwicklungsprozess, lassen sich grundsätzlich zwei Typen unterscheiden. Auf der einen Seite finden sich firmeneigene Umsetzungen, auf der anderen Seite Lösungen, die (meist kostenpflichtig) von externen Partnern bereitgestellt werden.
Vor allem große Konzerne und prominente Vorreiter dieser Programme, wie zum Beispiel Google [23], PayPal [18] und Microsoft [24], haben eigene Systeme implementiert. Dies lässt sich unter anderem mit der fehlenden Verfügbarkeit externer Dienstleister zum Startzeitpunkt der Programme begründen. Im Gegensatz dazu setzen gerade neue Einsteiger, wie zum Beispiel Qualcomm [25], Tesla [26] oder auch staatliche Institutionen wie das amerikanische Pentagon [27][28], auf die Angebote von Bug-Bounty-Portalen. Darüber hinaus gibt es bereits einige Firmen, wie zum Beispiel Yahoo, die zwar mit einem eigenen Programm starteten, dieses aber zu einem Portal eines Drittanbieters portierten [29].
Zwei populäre Beispiele für diese Anbieter sind die beiden Bug-Bounty-Plattformen HackerOne und Bugcrowd [30].
Dort können sowohl Firmen als auch Hacker Profile anlegen und die Plattformen als Basis ihrer Kommunikation nutzen. Während Hacker häufig auf mehreren Plattformen registriert sind, bieten Firmen gewöhnlicher Weise nur auf einer Plattform ihr Bug-Bounty-Programm an.
Für die Hacker ergeben sich durch solche Plattformen erhebliche Vorteile. Einerseits regeln die Betreiber der Plattformen den generellen Ablauf der Bug-Bounty-Programme und geben einen Rahmen für die spezifischen Modalitäten eines Programms. Damit ermöglichen sie eine bekannte und zuverlässige Struktur und geführte Kommunikation mit der Firma des gefundenen Bugs. Anderseits bieten sie die Möglichkeit ein einheitliches und öffentliches Profil (siehe Abbildung 3) zu präsentieren, mit dem bei diversen Gelegenheiten (Jobinterviews, Coaching Angebote etc.) geworben werden kann.
Aber auch für die Firmen liegen die Vorteile von Plattformen für Bug-Bounty-Programme auf der Hand. Einige der bereits genannten Bestandteile müssen nicht mehr selber gebaut, sondern nur in den eigenen, vorhandenen

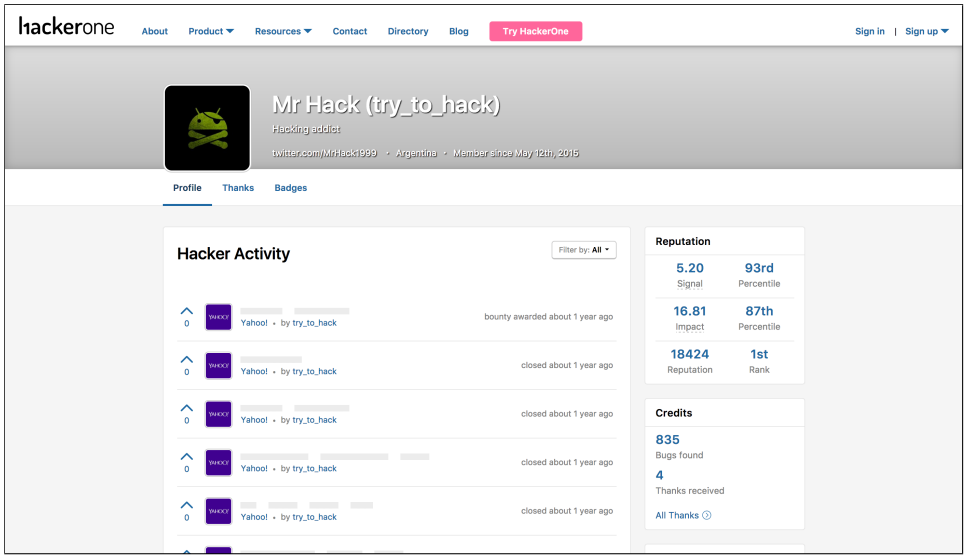


Abbildung 3: Das Nutzerprofil eines Hackers auf HackerOne

Softwarenentwicklungsprozess integriert werden. Damit wird eine wesentliche Komplexität an den Betreiber der Plattform abgegeben. Dieser kann den Betrieb des Programms kostengünstig anbieten, da sich die Anforderungen an Bug-Bounty-Programme unterschiedlicher Firmen kaum unterscheiden. Meist bieten Plattformen ein einfaches Verwaltungssystem für Bug-Bounty Meldungen, welches vermittelt, in welchem Zustand sich eine Meldung derzeit befindet. Zudem wird häufig die Möglichkeit geboten, die Meldung spätestens nach der Bug-Verifizierung in das eigene Aufgabensystem zu exportieren, sodass sie, mit allen Informationen aus der Plattform angereichert, dem gesamten Entwicklungsteam zur Verfügung steht. Entscheidet sich die Firma im Abschluss des Prozesses für eine monetäre Belohnung, wird auch diese von der Plattform übernommen. In diesem Fall wird die Summe meist an die Plattform gezahlt, die einen gewissen Prozentsatz als Provision einbehält und den Rest an den Hacker weiterleitet. HackerOne gibt diesen Wert beispielsweise mit 20% an [31]). Auf diese Art und Weise kann der Firma von der Bug-Bounty-Plattform eine formale Rechnung gestellt werden. Diese stellt eine reguläre Zahlungsverbindlichkeit dar, die ohne Probleme bedient werden kann. Soll die Sicherheitslücke publiziert werden, kann dies meist direkt auf der Plattform passieren. Hierbei stehen oftmals mehrere Varianten zur Verfügung, etwa eine Zusammenfassung, genaue Details zur Lücke oder sogar die Veröffentlichung aller Informationen inklusive des gesamten Gesprächsverlaufs. Bei all diesen Varianten ist es auch üblich einzelne Teile für die Publikation zu schwärzen.

5 Motivation und Determinanten

An einem Bug-Bounty-Programm sind mit Hackern und Produkt-Stakeholdern zwei Gruppen von Akteuren beteiligt. In diesem Kapitel sollen zunächst ihre jeweiligen, unterschiedlichen Motivationen herausgearbeitet und anschließend Determinanten abgeleitet werden. Nur unter Berücksichtigung dieser kann ein Bug-Bounty-Programm entwickelt werden, das den Ansprüchen beider Parteien genügt und eine fruchtbare Zusammenarbeit fördert. Sobald ein Programm jedoch nur die Bedürfnisse einer Akteursgruppe befriedigt, ist der Erfolg nicht mehr gewährleistet. Werden auf der einen Seite die Ansprüche der Hacker ignoriert, ist mit mangelnder Beteiligung zu rechnen, werden auf der anderen Seite die Ansprüche der Produkt-Stakeholder ignoriert, können Bug-Meldungen nicht ideal verarbeitet und in den Softwareentwicklungsprozess integriert werden.

5.1 Hacker

Um die Motivationen der Hacker differenziert betrachten zu können, ist es sinnvoll vorher eine übliche Kategorisierung einzuführen. So lassen sich Hacker grundsätzlich in drei Kategorien einordnen: White-Hat-, Gray-Hat- und Black-Hat-Hacker [32]. Während die zuletzt genannten entweder aus rein destruktiver, krimineller Motivation handeln und Schwächen der Software zu ihrem eigenen Vorteil ausnutzen, so handeln die Hacker der anderen beiden Kategorien aus intrinsischer oder legaler extrinsischer Motivation [33], wie etwa öffentlicher Reputation oder aus monetärem Interesse. Ihrer jeweiligen Motivation folgend informieren sie die verantwortlichen Produkt-Stakeholder umgehend über aufgedeckte Schwachstellen in deren Software. Der Unterschied zwischen den beiden letztgenannten Gruppen ist, dass White-Hat-Hacker explizit und offiziell zum Finden von Sicherheitslücken aufgefordert werden, wohingegen Grey-Hat-Hacker diesen Prozess aus eigener Motivation, also nicht zum Beispiel als Reaktion auf ein bestehendes Bug-Bounty-Programm einleiten.
Sofern Bug-Bounty-Programme eine monetäre Entlohnung vorsehen, bedienen Sie damit die Motivationen der White- und Grey-Hat-Hacker. Black Hat Hacker können auch erreicht werden, allerdings meistens nur, wenn die ausgelobte Summe den möglichen Verdienst auf dem Schwarzmarkt übersteigt [30][34]. Eine Lücke kann dort jedoch leicht die fünf- bis fünfundzwanzigfache Summe einbringen [20]. Ein weiterer Faktor für einen Black-Hat-Hacker kann auch das geringere Risiko sein, welches ihn dazu bewegt, selbst bei geringer, negativer Differenz der Auszahlungssumme den legalen Weg und damit das Bug-Bounty-Programm vorzuziehen. Dies bezieht sich etwa auf die tatsächliche Auszahlung der vereinbarten Belohnung und die ausbleibende Strafverfolgung, beziehungsweise das nicht mehr notwendige Verschleiern mittels diverser Anonymisierungsdienste sein. So gaben laut einer empirischen Stu-

die aus dem Jahr 2016 60% der Hacker an, Angst vor Strafverfolgung nach dem Veröffentlichen einer Sicherheitslücke zu haben. [35]
All drei Gruppen der Hacker haben eine sehr hohe Unsicherheit bezüglich der möglichen Entlohnung ihrer Arbeit. So steht die Auszahlungssumme im Rahmen eines Bug-Bounty-Programms nicht im Verhältnis zur geleisteten Arbeit, sondern zum Wert der Softwarelücke am Markt [36]. Dieser orientiert sich an dem potentiellen Schaden der Lücke und nicht am Aufwand diese zu finden. Das kann im Extremfall bei einer sehr gefährlichen Lücke, die mit wenig Arbeit aufzufinden war, einen sehr hohen Zeitlohn oder aber bei einer ungefährlichen Lücke, die nur mit viel Aufwand aufzufinden war, einen sehr niedrigen Zeitlohn bedeuten. Damit rentieren sich Bug-Bounty-Programme meist nicht für hochqualifizierte Softwareentwickler, da der hohe zeitliche Einsatz keinen Erfolg garantiert. Stattdessen nehmen zum Großteil selbst erlernte Hacker an solchen Programmen teil, die aufgrund der Wirtschaftslage derzeit keine anderen Jobperspektiven haben oder mit ihren sehr geringen technischen Kompetenzen für klassische Jobs in der Softwareentwicklung unterqualifiziert sind [21]. Dies bestätigen auch die jeweiligen Mitgliederumfragen aus dem Jahr 2016 der Plattformen HackerOne[37] und BugCrowd [38].
Zudem ist keinerlei Garantie auf Erfolg, also dem Finden von Sicherheitslücken, fgegeben. Natürlich steigt die Chance Bugs zu finden mit höheren Kompetenzen, jedoch ist es für Hacker schwer abzuschätzen, wo potentiell wertvolle Softwarelücken zu finden sein könnten. Des Weiteren ist das Risiko, dass ein anderer Hacker diese Lücke bereits gefunden hat, gerade bei Bug-Bounty-Programmes großer Firmen sehr hoch. Das liegt daran, dass es keinerlei Verzeichnis für Hacker gibt, in dem zu sehen ist, welche Sicherheitslücken zwar bereits gefunden aber noch nicht behoben wurden. Offensichtlich ist dies nicht möglich, da es in einem Bug-Bounty-Programm keine Einschränkung der Teilnehmer gibt und somit jeder, also auch ein potentieller Angreifer, Einsicht in alle existierenden Lücken hätte.
Die öffentliche Anerkennung stellt eine weitere Form der Motivation dar. So veröffentlichen die meisten Bug-Bounty-Programme, selbst wenn sie zum Inhalt der gefundenen Lücke keine Details nennen möchten, allein die Tatsache, dass ein bestimmter Hacker erfolgreich am Programm partizipiert hat. Dies bedeutet für den Hacker ein öffentliches Zugeständnis einer (namhaften) Firma in Bezug auf berufliche Kompetenz und Ethik und damit bessere Chancen auf dem Arbeitsmarkt [39]. Es ist daher für Hacker nicht unüblich die erfolgreichen Teilnahmen an Bug-Bounty-Programmen auf persönlichen Webpräsenzen oder in Lebensläufen zu nennen.

5.2 Produkt-Stakeholder

Die primäre Motivation für die Produkt-Stakeholder, ein Bug-Bounty-Programm zu betreiben, besteht darin keine, beziehungswiese keine bekannten,

sicherheitsrelevanten Bugs in der eigenen Software zu haben. Unter Verwendung eines Bug-Bounty-Programms können diese im Anschluss an Meldungen umgehend behoben werden und kommen so möglichst kurz oder gar nicht in Produktivumgebungen vor.
Ein (externer) Eingriff stellt immer auch eine potentielle Schwächung in den Softwareentwicklungsprozess dar. Es bedarf daher einer sorgfältigen Planung und strukturierten Durchführung, damit die Softwareentwicklung nicht verlangsamt oder gar behindert wird, sondern im Idealfall sogar eine Bereicherung des Prozesses erfolgt [40]. Wird ein Bug-Bounty-Programm ineffizient implementiert, ergeben sich zum einen für das Team unklare Verantwortungen die einzelnen Aufgaben und Rollen eines Programms betreffend und zum anderen durch Aufgaben und Planänderungen irreguläre Störungen. Diese können je nach Softwareentwicklungsprozess zu unnötigem Verlust von Zeit und Ressourcen führen. Um dies zu verhindern sollten die Aufgabenbereiche sinnvoll auf Mitglieder des Teams verteilt werden. Darüber hinaus sollten Aufgaben vom Prüfen bis hin zu Codeänderungen als reguläre Aufgabe innerhalb des Prozesses implementiert werden und so im regulären Prozess abgearbeitet werden können.
Da es sich bei den meisten Softwareentwicklungsabteilungen um Teile wirtschaftlicher Einheiten handelt, gilt hier neben anderen Faktoren an erster Stelle das Prinzip der Wirtschaftlichkeit. Man könnte argumentieren, dass allein die Summe aller nichtwirtschaftlichen Effekte die Ausgaben eines Bug-Bounty-Programms rechtfertigt. Eine solche Annahme geht jedoch davon aus, dass ein Bug-Bounty-Programm im Vergleich zum Einsatz festangestellter Sicherheitsforscher keine ökonomischen Vorteile, eventuell sogar Nachteile, mit sich bringt. Eben dies ist aber einer empirischen Untersuchung zufolge nicht der Fall [41]; der Einsatz eines solchen Programmes bietet nämlich mit hoher Wahrscheinlichkeit Kostenersparnisse im Bereich informationstechnologischer Sicherheit. Die durch das Bug-Bounty-Programm ausgelösten Effekte auf andere Bereiche verändern diese verbesserte wirtschaftliche Ausgangslage nachträglich positiv (oder negativ).
Des Weiteren kann der Betrieb eines Bug-Bounty-Programms als Marketingmaßnahme gesehen werden. Einerseits stellt sich die Firma hier auf einer weiteren Plattform dar, die öffentlich einsehbar ist, und bietet damit Werbefläche und Kontaktpunkt, anderseits kann auch davon ausgegangen werden, dass diese Partizipation an einem Verfahren eine veränderte öffentliche Meinung über das Unternehmen hervorruft. Hierfür wäre zu klären, wie die Öffentlichkeit den Betrieb eines Programms wahrnimmt. In der Vergangenheit gab es vereinzelt kritische Stimmen zum Betrieb eines solchen Programms in Bezug auf die Ethik und Motivation der betreibenden Firmen [42]. Seit einiger Zeit herrscht sowohl fast in der gesamten fachnahen Welt Konsens über den Vorteil solcher Programme [43][44][45] als auch in der Öffentlichkeit. So ergab eine Studie aus dem Jahr 2017 von Kaspersky, dass (zumindest im amerikanischen Raum) 22% der Befragten beim Kauf eines Produkts ei-

ne Firma bevorzugen würden, die bei der Entwicklung ihrer Software mit Hackern zusammengearbeitet hat [46]. Durch die positive Wahrnehmung im fachnahen Raum kann zudem davon ausgegangen werden, dass dies auch dem Markenaufbau als Arbeitgeber hilft und potentiellen Bewerbern durch den als fortschrittlich und transparent angesehen Schritt imponiert [47].
Sofern sich die Betreiber eines Bug-Bounty-Programms dazu entscheiden eine Lücke nach Meldung und erfolgreicher Behebung zu publizieren, beziehungsweise einer Publizierung durch den Hacker zuzustimmen, erfolgt dies unter anderem, um die eben genannten marketingtechnischen Vorteile zu unterstützen. Eine weitere Motivation kann für die Firma darin bestehen, sofern eine große Gefahr von der Lücke ausgeht, diese zu Publizieren um Nutzer dazu zu bringen die Applikation upzudaten. Dies ist aber umstritten, da gerade bei clientbasierter Software durch das Publizieren ein hohes Risiko eingegangen wird. So war es Hackern zum Beispiel nur durch das Veröffentlichen der ShellShock-Lücke möglich ein riesiges Botnet aufzubauen [48]. Symantec geht bei einzelnen Lücken von einer Erhöhung der Ausnutzungsrate von bis zum Hunderttausendfachem aus [49].
Bei der Überlegung, ob und in welche Höhe monetäre Auszahlungen angeboten werden sollte, spielt es für Unternehmen eine große Rolle, diese in Relation zu dem potentiellen Schaden durch das Ausnutzen der Sicherheitslücke oder auch nur das Bekanntwerden dieser zu setzen. Hier kann davon ausgegangen werden, dass selbst bei den hohen maximalen Auszahlungssummen von Firmen wie Uber, DropBox und Twitter im fünfstelligen Bereich (USD) [50][51][52], die Höhe des ausgezahlten Betrags noch weit unter dem potentiellen Schaden liegt. Belohnungen sollten mit Bedacht und gerade bei größeren Firmen in angemessener Höhe ausgewählt werden. Beim "T-Shirt Gate" von Yahoo etwa entschied sich das Unternehmen, Übersendern von Sicherheitslücken als Dankeschön ein Yahoo-T-Shirt und, bei mehr als einer positiven Meldung pro Person, Geschenkegutscheine für den Yahoo-Fan-Shop im Wert von $12.50 zu verschicken. Damit löste der Konzern mediale Aufregung und Ärger in der Internet-Community aus [20][53].
Betrachtet man das Spektrum verschiedener Hacker, die Softwarelücken finden und potentiell an das Unternehmen oder andere Parteien weitergeben können, so ist ein Bug-Bounty-Programm mit Auszahlungen eine einzigartige Möglichkeit, um alle drei Hackertypen gleichzeitig anzusprechen. White-Hat-Hacker nehmen per se erst nach der Implementierung eines Bug-Bounty-Programm und der damit verbundenen Freigabe die Suche nach Sicherheitslücken in der Software auf. Grey-Hat-Hacker suchen möglicherweise sowieso nach Bugs im Produkt, haben aber Probleme die Informationen an die richtige Stelle des Unternehmens weiterzuleiten. Black-Hat-Hacker hingegen haben neue extrinsische Motivationen, mit dem Unternehmen zu kooperieren.

6 Implementierung

In Reaktion auf den in Kapitel 2.3 beschriebenen Vorfall herrschte innerhalb von eBay Kleinanzeigen Konsens bezüglich der Notwendigkeit der baldigen Implementierung eines Bug-Bounty-Programms. Im Folgenden werden die tragenden Entscheidungen dokumentiert, die im Verlauf der Implementierung getroffen wurden.

6.1 Art der Implementierung

Als erstes wurde die aktuellen Situation festgehalten, und es wurde evaluiert, welche Möglichkeiten es zur Inbetriebnahme eines Bug-Bounty-Programms es geben würde. Nachdem Kontakt mit zuständigen Mitgliedern der eCG-InfoSec Abteilung aufgenommen worden war, informierten diese eBay Kleinanzeigen darüber, dass für alle Entitäten der eCG ein Rahmenvertrag mit der Plattform HackerOne bestand. Dieser wurde zu diesem Zeitpunkt nur von Markplaats[8] genutzt, hatte aber freie Kapazität für weitere eCG-Produkte. Da auch die Schwesterplattform über positive Erfahrungen berichtete, wurde entschieden, das Bug-Bounty-Programm in Kooperation mit HackerOne auf deren Plattform zu eröffnen.

6.2 Aufbau

Auf Basis der bereits vorgenommen Evaluierung der Bestandteile eines Bug-Bounty-Programms (siehe Kapitel 4.1) wurde Schritt für Schritt überlegt, wie eine ideale Umsetzung für eBay Kleinanzeigen aussehen könnte.

Kommunikation über Vorhandensein

Durch die Entscheidung mit einer externen Plattform zu kooperieren, bot sich an, diese als Basis der Kommunikation über das Vorhandensein zu verwenden. Daher lag im Folgenden der Schwerpunkt darauf, die Modalitäten des Programms zu definieren, klar zu kommunizieren und das Programm leicht erreichbar zu machen. Dabei wurde sich inhaltlich an den Modalitäten und Programmtexten anderer großer Firmen orientiert. So entstand ein verschriftlichtes Programm, welches dem Corporate-Image und den Vorstellungen eBay Kleinanzeigens entsprach. Es wurde als wichtiges Kriterium angesehen, dass das Bug-Bounty-Programm über diverse Suchmaschinen einfach zu finden sein sollte. Aus diesem Grund wurde beschlossen eine auf der Startseite verlinkte, statische Seite hinter der eBay Kleinanzeigen Domain zu platzieren, die das Bug-Bounty-Programm präsentierte und auf das HackerOne-Profil verwies. Dies ist trotz der SEO-starken HackerOne-Website

[8] Markplaats.nl ist Tochter der eBay Classifieds Group und das holländische Pendant zu eBay Kleinanzeigen.

gerade bei eBay Kleinanzeigen wichtig, da das Kleinanzeigenportal, wie die meisten Online-Marktplätze, eine generische SEO-Implementierung vorweist, die jeden Suchbegriff zusammen mit dem Produktnamen generisch im eignen Sortiment suchen lässt. Die statische Seite enthält keine Nennung etwaiger Modalitäten des Programms, um die Gefahr asynchroner Informationen zu umgehen.

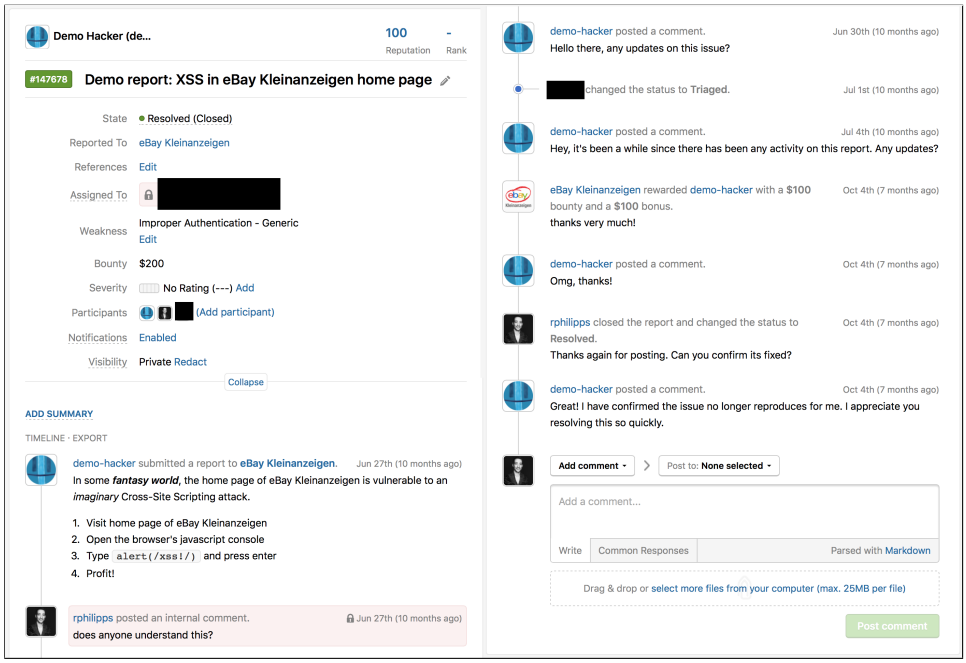


Abbildung 4: Ein exemplarischer Konversationsthread im Demo-Modus auf HackerOne

Schnittstelle für Kommunikation

Die Schwesterplattform Marktplaats berichtete, dass das zusätzliche Einrichten einer Email-Adresse mit hohem Spamaufkommen verbunden wäre und damit einen weiteren Prozess mit neuen Verantwortungen kreieren würde. Deswegen wurde entschieden, ausschließlich Bug-Meldungen via HackerOne zu erlauben. Dennoch wurde eine Email-Adresse für Rückfragen von Hackern eingerichtet. Diese wird jedoch, wie auch auf dem HackerOne-Profil kommuniziert, nur sporadisch eingesehen. Bug-Meldungen werden dort im Notfall zwar angenommen, da hier die Sicherheit der Applikation über der Souveränität des Prozesses steht; eine Entlohnung und klassische Partizipation am Bug-Bounty-Programm ist aber für Hacker in diesem Falle ausgeschlossen.

Bei HackerOne haben Hacker die Möglichkeit eine Bug-Meldung mittels eines Web-Formulars aufzugeben. Ab diesem Zeitpunkt können sie mit der Firma auf der Plattform in einem Konversations-Thread über

eine Meldung kommunizieren (siehe Abbildung 4). Im Verlauf der Kommunikation ist meist ein bestimmter Mitarbeiter für diese Kommunikation zuständig, theoretisch kann die Konversation aber mit beliebig vielen Teammitgliedern des Unternehmens erfolgen.
Durch die Verwendung des HackerOne-Angebots und der Kompatibilität mit dem IT-Eco-System eBay Kleinanzeigens ergaben sich diverse Möglichkeiten, um eine Integration in bestehende Prozesse und Systeme vorzunehmen. So wird die Security-Guild sowohl über neue eingehende Meldungen als auch über jegliche Statusänderungen bereits erfolgter Meldungen automatisch via Instant-Messaging benachrichtigt. Jedes Mitglied wird dadurch aktiv über wichtige Veränderungen informiert und muss nicht permanent eigenständig überprüfen, ob es neue Ereignisse gab, die eine Reaktion erfordern.

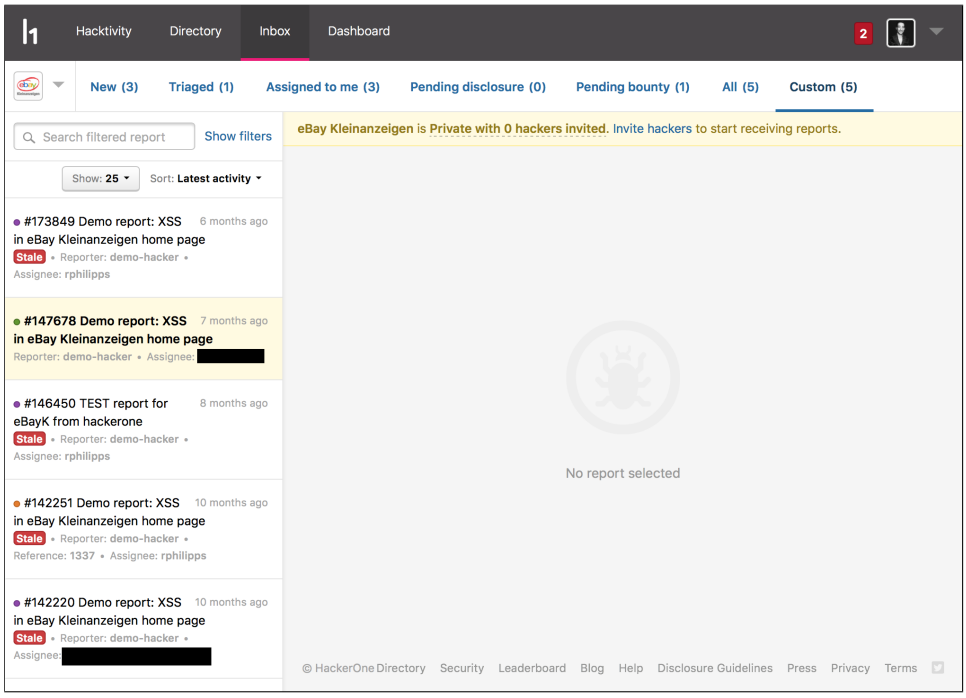


Abbildung 5: Das Board für Bug-Meldungen von eBay Kleinanezigen im Demo-Modus auf HackerOne

Bug-Entgegennahme

Nach Eingang einer neuen Bug-Meldung wird von einem beliebigen Mitglied der Security-Guild überprüft, ob es sich um eine Sicherheitslücke handeln könnte und die formalen Kriterien erfüllt sind. Auf dieser Basis wird entschieden, ob die Bug-Meldung angenommen, abgelehnt oder der Hacker um weitere Informationen gebeten wird. HackerOne bietet für eingehende Bug-Meldungen ein internes Board (Siehe Ab-

bildung 5), auf dem die Tickets bewegt werden können, sodass alle Teammitglieder sich jederzeit über den aktuellen Status aller Meldungen informieren können.
Dadurch, dass HackerOne anonymisierten Zugriff auf eine Vielzahl an Bug-Meldungen und deren Verarbeitung in den Bug-Bounty-Programmen hat, stellt die Plattform Bug-Bounty-Betreibern eine Vielzahl von Möglichkeiten zur Verfügung, das hohe Aufkommen von Spam-Anfragen zu verringern. Die Plattform bietet automatisierte Spam-Erkennungstools, die entscheiden, ob eine Anfrage an das Unternehmen weitergegeben werden soll. So geben zum Beispiel Uber [54] oder Slack [55] an, dass die Falschmeldungen bei ihren Programmen auf HackerOne nur ungefähr fünf Sechstel des Aufkommens ausmachen. Des Weiteren kann jede Firma genau festlegen, welche Informationen in eine Meldung inkludiert werden müssen. Diese werden dann auf HackerOne vom Hacker direkt im Meldeformular abgefragt. Dadurch lassen sich Falschmeldungen zwar nicht verhindern, es ist aber gerade mit den Erfahrungen von Marktplaats davon auszugehen, dass deren Anzahl zumindest stark verringert werden kann.

Bug-Verifizierung
Erfüllt eine Bug-Meldung alle formalen Kriterien, wird von einem beliebigen Mitglied der Security-Guild überprüft, ob sich der Bug wie angegeben nachstellen lässt und nicht in externen sowie aktuellen Bibliotheken befindet. Sofern die verwendete Version der externen Bibliothek mehr als sechzig Tage hinter der Veröffentlichung liegt, wurde entschieden, dieses auch als (zu entlohnende) Sicherheitslücke anzuerkennen. In dem Fall wäre für eBay Kleinanzeigen Handlungsbedarf gegeben und damit die Meldung sinnvoll, um die Sicherheit der Applikation zu gewährleisten.

Vorfallüberprüfung
Nach der erfolgreichen Verifizierung eines Bugs wird dieser direkt vom bearbeitenden Entwickler auf das Kanban-Board exportiert. Hierzu bietet HackerOne eine Schnittstelle an, die während der Vorbereitung des Bug-Bounty-Programms integriert wurde. Damit kann eine Bug-Meldung auf HackerOne direkt mit einem Mausklick als neues Ticket in das verwendete Jira-System migriert werden. Hierbei werden alle Informationen wie Rückfragen und die genaue Beschreibung inklusive Screenshots behalten, sodass an dieser Stelle keine zusätzliche manuelle Arbeit notwendig ist. Auf dem Kanban-Board wird das Ticket nach der Zero-Security-Bug-Policy (siehe Kapitel 2.1.2) umgehend ganz oben in der “ToDo”-Spalte platziert und steht für alle Entwickler zur Verfügung. Die darin beschriebene Aufgabe besteht zum Großteil aus dem Folgepunkt “Softwareanpassung”, aber immer auch darin, vorher zu

überprüfen, ob der Bug in der Vergangenheit bereits von Angreifern ausgenutzt wurde. Ist letzteres der Fall, werden mit der Security-Guild die nächsten Schritte besprochen und gegebenfalls ein anderes derzeit bearbeitetes Ticket pausiert, um eine schnelle Abarbeitung des Security-Bugs zu ermöglichen.

Softwareanpassung

Das bereits in das Jira-System und auf der "ToDo"-Spalte platzierte Ticket kann von einem beliebigen Entwickler genommen werden, sobald dieser freie Kapazität zur Bearbeitung hat. Wie bei einem gewöhnlichen Task und schon in Kapitel 2.1.1 beschrieben, wird das Ticket also zuerst entwickelt, dann der Code reviewed und zum Abschluss auf einer Testumgebung qualitätsgesichert. Dadurch kann gewährleistet werden, dass die aus der Bug-Meldung entstandene Anforderung und der Fortschritt dokumentiert sind und dadurch selbst bei Ausfall des verantwortlichen Entwicklers eine zügige Abarbeitung gegeben ist.

Fix-Validierung

In den wöchentlichen Meetings der Security-Guild wird der Stand aller aktuell offenen Security-Tickets anhand des Boards auf HackerOne besprochen. Dabei wird überprüft, ob eine Änderung seitens der Entwicklung erfolgte oder sogar der Fix zu einem Ticket ins Produktivsystem eingespielt wurde. In dem Fall, wird der Hacker erneut via HackerOne kontaktiert und gebeten das Beheben der Lücke zu bestätigen. Dies ist meist nur eine Formalie, da dieses durch die Qualitätssicherung bereits überprüft sein sollte.

Belohnungs- und Publikationsfestsetzung

Für die Festsetzung der Belohnung wurde von HackerOne empfohlen, eine Tabelle vorzugeben, die klar definiert, für welchen Typ von Sicherheitslücke welche konkrete Summe ausgezahlt wird. Dies würde laut HackerOne Komplikationen auf beiden Seiten und Unzufriedenheit auf Seiten der Hacker vorbeugen. Es wurde entschieden sich an der internen InfoSec-Klassifizierung von Sicherheitslücken in vier verschiedene Stärken zu orientieren[9] und damit vier verschiedenen Belohnungsgrößen anzubieten. Somit soll Klarheit geschaffen werden und auch für noch nicht bedachte oder neue Angriffsszenarien eine Größenordnung der Auszahlung gegeben sein. Die Kategorisierung können die Hacker mittels des offenen Standards eigenständig bereits im Vorhinein vornehmen. Die daraus ergebende Summe wird umgehend via HackerOne freigegeben, sobald der Fix vom Hacker validiert wurde.

Es wurde sich gegen das Zusenden von Merchandise entschieden, da es

[9] Bei eBay gibt es für die Klassifizierung von Sicherheitslücken vier Stufen von P1 (maximal kritisch) bis zu P4 (minimal kritisch). Diese basieren auf bestimmten Intervallen des Common-Vulnerability-Scoring-Systems (CVSS) [56].

einen zu großen Aufwand bedeuten würde, nationale Versandbesonderheiten zu recherchieren. Zudem bedeutet das Packen der Pakete einen unverhältnismäßig großen Aufwand für jede erfolgreiche Meldung und bietet keinerlei Optimierungspotential zur besseren Skalierung.
Sofern keine direkte Anfrage zur Publikation vom Hacker erfolgt, wurde beschlossen, die Sicherheitslücke nicht eigenständig zu veröffentlichen. Dieses lässt sich durch die Sorge über negative öffentliche Wahrnehmung in Reaktion auf Sicherheitslücken im Produkt erklären. Gerade bei nicht ausgenutzten Lücken stellt dies nach Ansicht des Unternehmens ein unverhältnismäßig hohes Risiko dar. Zudem entstünde bei jeder Publikation hoher administrativer Aufwand, da diverse Personen und -gruppen beteiligt sein müssten. So ist es die Regel, dass vor Veröffentlichungen einerseits der CTO von eBay Kleinanzeigen eine Freigabe erteilt, die Marketingabteilung konsultiert wird und zum Abschluss eine Überprüfung durch die Rechtsabteilung erfolgt. Sofern ein Hacker eine Veröffentlichung wünscht, wird dieser Prozess aber angestoßen, zumal sich eine Veröffentlichung auf einem Blog des Hackers oder zum Beispiel in sozialen Netzwerken ohnehin kaum verhindern ließe.

6.3 Kompetenzmatrix

Bei Betrachtung der theoretischen Kompetenzmatrix (siehe Kapitel 4.2) fällt auf, dass ursprünglich eine Unterteilung zu Personen mit Kommunikationskompetenz erfolgte. Diese Rolle übernehmen bei eBay Kleinanzeigen Mitglieder der Security-Guild, und führen damit die initiale sowie die abschließende Kommunikation. Für die Publikationsmöglichkeit wird aber auf Ressourcen innerhalb des Marketings zurückgegriffen.

	Marketing	Security-Guild	SWE (+P) (+C)
Bug-Entgegennahme		x	
Bug-Verifizierung		x	
Vorfallüberprüfung			x
Softwareanpassung			x
Fix-Validierung		x	
Belohnungs- und Publikationsfestsetzung	x	x	

Tabelle 2: Angewandte Kompetenzmatrix für eBay Kleinanzeigen.

Für die Art und Weise der Kommunikation wurden publizierte Konversationsverläufe auf HackerOne sowie die Erfahrungen aus Marktplaats verwendet. Hieraus ergab sich eine kurze Leitlinie die vor allem auf folgende Punkte

Wert legt:

- Die Kommunikation sollte immer offen und freundlich sein aber stets bestimmt. Es sollte immer eindeutig kommuniziert werden, dass Tempo und Herangehensweise von der Firma bestimmt wird.
- Es sollte darauf geachtet werden, die Hacker nicht zu lange im Unklaren zu lassen. Dies erfordert eine schnelle Rückmeldung auf die initiale Bug-Meldung und zügige Beantwortung von Rückfragen. Zudem wurde beschlossen bei langen Entwicklungszeiten zwischenzeitlich die Hacker darüber zu informieren, dass weiterhin am Bug gearbeitet wird.

Da bei eBay ein System des absoluten Vertrauens in die Mitglieder der Softwareentwicklung herrscht, hat jeder Entwickler von eBay Kleinanzeigen vollen Zugriff zum gesamten Programmcode sowie allen Produktivsystemen. Daraus ergibt sich, dass auf Basis der gleichen Rechte aller Entwickler keine Unterscheidung in den Punkten Bug-Verifizierung, Vorfallüberprüfung und Softwareanpassung notwendig ist. Nur weil die Bug-Verifizierung nicht in den regulären Softwareentwicklungsprozess integriert wurde, wird dieser Schritt nicht vom ganzen Team der Softwareentwickler, sondern von Mitgliedern der Security-Guild übernommen. Vorfallüberprüfung und Softwareanpassung werden dann aber regulär von gemeinen Entwicklern vorgenommen (die aber natürlich auch Mitglied in der Security-Guild sein könnten). Hieraus ergibt sich die in Tabelle 2 dargestellte für eBay Kleinanzeigen angewandte Kompetenzmatrix.

6.4 Startvorbereitungen

Da durch eBay-InfoSec, in langjährig geplanten Rahmenverträgen sowieso Penetration-Tests in regelmäßigen Abständen für die eBay Kleinanzeigen Plattform vorgesehen sind, wurde entschieden die Durchführung eines solchen, baldigen Tests abzuwarten. Mit dem Start des Bug-Bounty-Programms wird begonnen, sobald alle gefundenen Lücken behoben worden sind. Dies erscheint sinnvoll, da so nicht für eine bereits gebuchte Leistung zusätzlich Hacker im Rahmen des Bug-Bounty-Programms entlohnt werden müssen.
Ein großer Vorteil, der sich durch HackerOne bietet, ist die Möglichkeit eines Soft-Launchs. HackerOne bietet an, Bug-Bounty-Programme anfänglich nur sehr wenigen, aber bewährten Hackern zur Verfügung zu stellen. Danach wird das Programm langsam immer weiteren Mitgliedern der Plattform zur Verfügung gestellt, bis es zum Abschluss für alle Nutzer öffentlich einsehbar und benutzbar ist. Die betreibende Firma des Bug-Bounty-Programms kann langsam auf das Arbeitspensum vorbereitet werden. So wird den Teams die Möglichkeit gegeben, mögliche Schwächen im Prozess zu identifizieren und zu beheben, bevor das Programm von einer Großzahl von Hackern genutzt wird und damit potentiell Schaden im Softwareentwicklungsprozess anrichten kann.

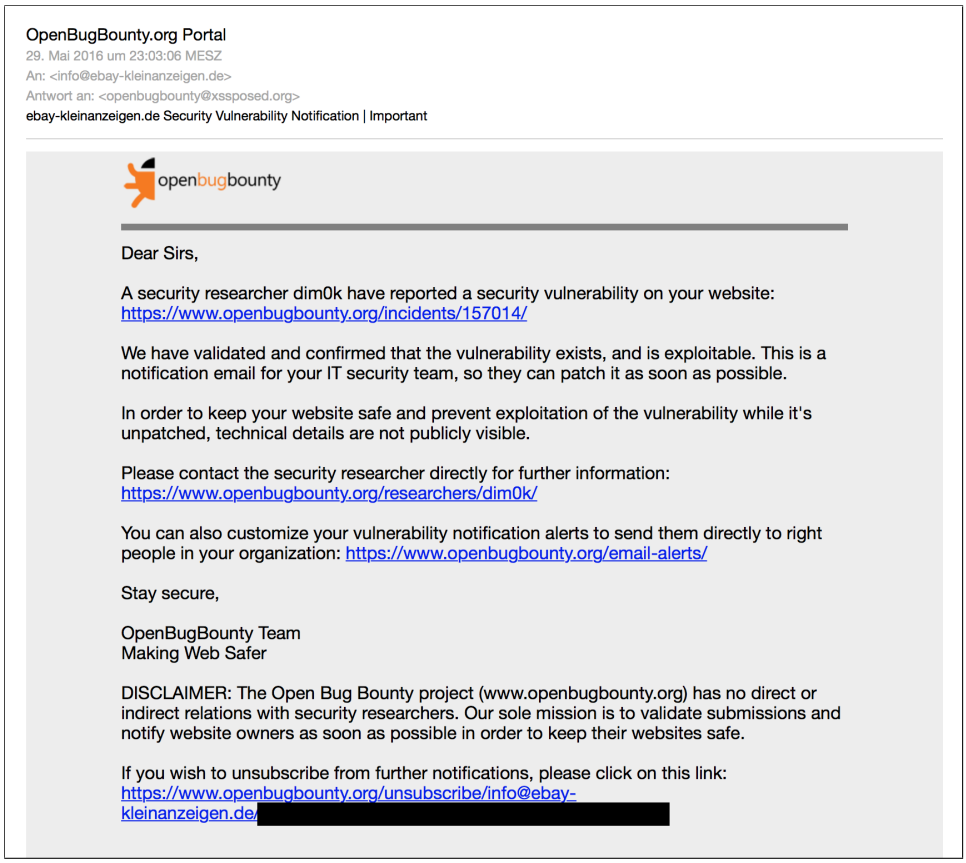
7 Diskussion

Ziel der vorliegenden Arbeit war es, ein allgemeines Modell zum Aufbau von Bug-Bounty-Programmen zu entwickeln und eine Implementierung in einen vorhandenen Softwareentwicklungsprozess zu dokumentieren. Als Fallbeispiel hierfür diente die Implementierung bei eBay Kleinanzeigen, einem deutschen Online-Portal für Kleinanzeigen. Nach einem Vorfall wurde dort festgestellt, dass keine Prozesse vorhanden waren, um mit Bug-Meldungen umzugehen und so der Bedarf für ein Bug-Bounty-Programm gegeben war.
Dazu wurde beschlossen, das den Bug-Bounty-Programmen zugrundeliegende theoretische Konzept aufzuarbeiten, um Wirkmechanismen abzuleiten und die Erkenntnisse darüber zur Implementierung des Programms bei eBay Kleinanzeigen zu verwenden. Auf Basis wissenschaftlicher Publikationen und Analyse der Abläufe verschiedener Bug-Bounty-Programme wurde ein achtstufiger Aufbau konstruiert, der die prototypischen Bestandteile eines Programms aufzeigt. Daraus wurde eine Kompetenzmatrix abgeleitet, mit der die Verantwortungsverteilung der Bestandteile auf verschiedene Personengruppen dargestellt werden kann.
Anschließend wurden die unterschiedlichen Motivationen der beteiligten Akteure analysiert und die resultierenden Determinanten dargestellt, die notwendig sind, um alle beteiligten Parteien zufriedenzustellen.
Die abschließende Implementierung orientierte sich am theoretischen Aufbau. Das entwickelte theoretische Modell konnte im vorliegenden Fall sehr gut angewendet werden. Es wäre jedoch zu verifizieren, inwiefern es ohne größere Veränderung auch auf Firmen mit anderer Größe, Struktur und mit anderem Softwareentwicklungsprozess übertragbar ist. Der Startzeitpunkt des Bug-Bounty-Programms liegt weniger Tage nach der Abgabe dieser Arbeit. Daher lässt sich noch kein Urteil über den Erfolg oder Misserfolg des Programms fällen.
Bei eBay Kleinanzeigen sind mit Einbau des Bug-Bounty-Programms in Zukunft einkommende Meldungen zu Sicherheitslücken keine Besonderheit mehr, sondern Teil des regulären Softwareentwicklungsprozesses. Während des Erstellen dieser Arbeit entschieden sich auch diverse andere Firmen wie Intel [57] und Nintendo [58] für den Start von Bug-Bounty-Programmen. Dies unterstreicht die Relevanz solcher Programme weiter. Zudem steigt der Druck für große Unternehmen mit jedem weiteren Konzern, der im Gegensatz zu ihnen ein solches Programm anbietet. Voraussichtlich wird es in den nächsten Jahren für jede größere Firma eine Selbstverständlichkeit sein, ein gut funktionierendes Bug-Bounty-Programm mit monetären Entlohnungen zu betreiben. Damit wird ein Bug-Bounty-Programm in naher Zukunft fester Bestandteil eines modernen und gesunden Softwareentwicklungsprozesses sein. Zukünftig wird das Nichtvorhandensein eines solchen Programms im öffentlichen Bild vermutlich auf Unverständnis stoßen und als vermeidbare Gefahr für Daten, Nutzer und Software angesehen werden.

Anhang

A Kommuikationsverlauf mit Hacker

A.1 openbugbounty an eBay Kleinanzeigen | 29.05.2016 - 23:03



OpenBugBounty.org Portal
29. Mai 2016 um 23:03:06 MESZ
An: <info@ebay-kleinanzeigen.de>
Antwort an: <openbugbounty@xssposed.org>
ebay-kleinanzeigen.de Security Vulnerability Notification | Important

openbugbounty

Dear Sirs,

A security researcher dim0k have reported a security vulnerability on your website:
https://www.openbugbounty.org/incidents/157014/

We have validated and confirmed that the vulnerability exists, and is exploitable. This is a notification email for your IT security team, so they can patch it as soon as possible.

In order to keep your website safe and prevent exploitation of the vulnerability while it's unpatched, technical details are not publicly visible.

Please contact the security researcher directly for further information:
https://www.openbugbounty.org/researchers/dim0k/

You can also customize your vulnerability notification alerts to send them directly to right people in your organization: https://www.openbugbounty.org/email-alerts/

Stay secure,

OpenBugBounty Team
Making Web Safer

DISCLAIMER: The Open Bug Bounty project (www.openbugbounty.org) has no direct or indirect relations with security researchers. Our sole mission is to validate submissions and notify website owners as soon as possible in order to keep their websites safe.

If you wish to unsubscribe from further notifications, please click on this link:
https://www.openbugbounty.org/unsubscribe/info@ebay-kleinanzeigen.de/[redacted]

A.2 rphilipps@ebay.com an dim0k | 30.05.2016 - 17:02

Hi dim0k, my name is Robert and I am part of the ebay-kleinanzeigen.de development team here at eBay.

Yesterday, we were contacted by openbugbounty.org that you have found a XSS vulnerability on our site ebay-kleinanzeigen.de.

If that is the case, we would like to thank you for finding that vulnerability and especially for your professional behavior of not disclosing any details to the public.

For us, the safety of our users is always our #1 priority, so we would like to verify and fix that issue as soon as possible. Therefore we would like you to inform us about the vulnerability details.

Best from Berlin,
Robert Philipps

A.3 dim0k an rphilipps@ebay.com | 30.05.2016 - 17:11

Hi Robert,

I will send you the details of the vulnerability, eBay have a bugbounty program ?

A.4 dim0k an rphilipps@ebay.com | 30.05.2016 - 17:26

[In dieser Mail wird von dim0k kurz mit Hilfe von Screenshots das Ausnutzen der Sicherheitslücke erklärt.]

A.5 rphilipps@ebay.com an dim0k | 30.05.2016 - 18:15

Hi Dmitry[10],
thank you very much for your fast reply and your open disclosure of our security vulnerability!

Our development team is currently looking into that issue and we will contact you asap when there is a fix.

We sadly do not have a bug bounty program currently, but we are in the process of establishing one with the hackerone.com platform. We think that everyone who helps making our sites more secure should get a fair reward for that.
But because these are currently only future plans, I am not sure if there is any possibility for us right now to pay you any bug bounty...
I will contact you in the next days and will inform you regarding any possible rewards.

Of course we are happy to write something on your profile on openbugbounty.org in the "Recommendations and Acknowledgements“ area, if you would like us to do that.

Thanks again for your great work!

Best,
Robert

A.6 dim0k an rphilipps@ebay.com | 30.05.2016 - 18:31

Hi Robert,

Thanks, recommended on my page is a good idea.

[10] Da der Hacker auf einer Email Antwortete, die den Namen “Dmitry” als Absender trug, wurde zu diesem Namen gewechselt.

A.7 rphilipps@ebay.com an dim0k | 01.06.2016 - 23:19

Hi Dmitry,
thanks for your patience.

We have verified your report, have been able to reproduce the XSS exploit and as of now we have fixed the issue and the vulnerability is not existent anymore.

The team responsible for bug bounty rewards here et eBay informed us that there is sadly no monetary reward for non-persistent XSS exploits, but as promised we wrote a longer recommendation on your openbugbounty.org page.
We would also like to send you a small eBay Kleinanzeigen surprise package as a small token of appreciation. For that I would need a post address where we can deliver the package to.

Again, thank you very much for your great work, it was a pleasure working with you!
You made ebay-kleinanzeigen.de more secure for over 15 million unique users per month and you can be proud of that!

Best from Berlin,
Robert Philipps

A.8 dim0k an rphilipps@ebay.com | 02.06.2016 - 06:15

Hi Robert,

Thanks for a bounty !

My post address:

[Adresse des Hackers]

A.9 rphilipps@ebay.com an dim0k | 20.06.2016 - 14:58

Hi Dmitry,
the package is underway to you!

Could you please mark the issue as resolved on openbugbounty.org?

Thanks a lot!

Best from Berlin,
Robert

A.10 dim0k an rphilipps@ebay.com | 21.06.2016 - 17:24

Hi Robert,

I confirm the vulnerability fixed.
https://www.openbugbounty.org/incidents/157014/

Thanks for a bounty, I'll get back to you when receiving the package.

A.11 dim0k an rphilipps@ebay.com | 27.06.2016 - 19:16

Hi Robert,

The package arrived, customs require me to send a document confirming your part (original or a screenshot) please send us.

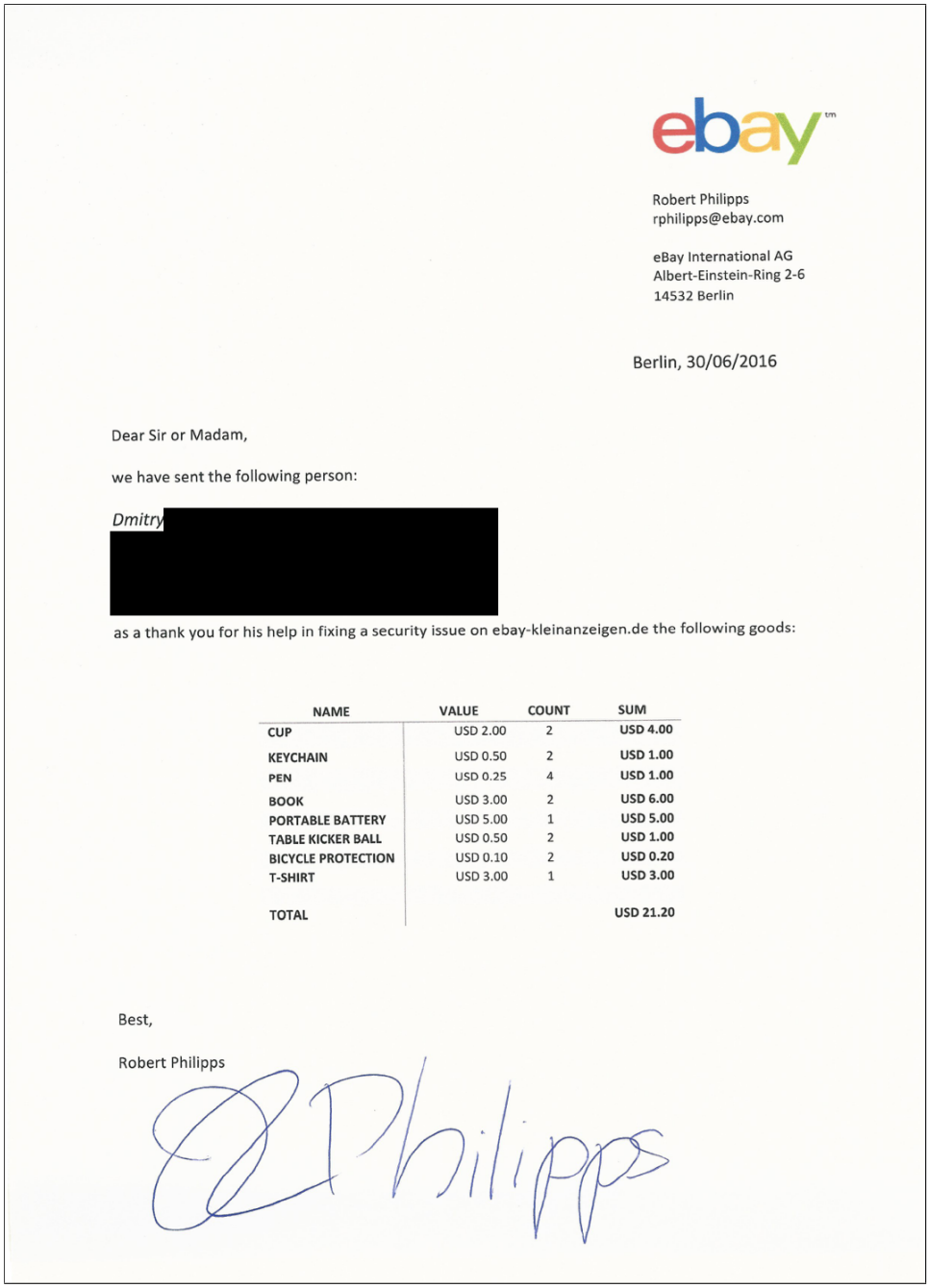
A.12 rphilipps@ebay.com an dim0k | 30.06.2016 - 13:29

Hi Dmitry,
sure thing!

A have attached a scan. If you need anything else to retrieve the package, please just tell me!

Best,
Robert

A.12 rphilipps@ebay.com an dim0k | 30.06.2016 - 13:29



Robert Philipps
rphilipps@ebay.com

eBay International AG
Albert-Einstein-Ring 2-6
14532 Berlin

Berlin, 30/06/2016

Dear Sir or Madam,

we have sent the following person:

Dmitry [redacted]

as a thank you for his help in fixing a security issue on ebay-kleinanzeigen.de the following goods:

NAME	VALUE	COUNT	SUM
CUP	USD 2.00	2	USD 4.00
KEYCHAIN	USD 0.50	2	USD 1.00
PEN	USD 0.25	4	USD 1.00
BOOK	USD 3.00	2	USD 6.00
PORTABLE BATTERY	USD 5.00	1	USD 5.00
TABLE KICKER BALL	USD 0.50	2	USD 1.00
BICYCLE PROTECTION	USD 0.10	2	USD 0.20
T-SHIRT	USD 3.00	1	USD 3.00
TOTAL			USD 21.20

Best,

Robert Philipps

A.13 rphilipps@ebay.com an dim0k | 19.07.2016 - 17:37

Hi Dmitry,
were you able to retrieve the package?

Because our postal service informed us that it could not be delivered... is that right?

Best,
Robert

A.14 dim0k an rphilipps@ebay.com | 27.06.2016 - 18:12

Hi Robert,

a few minutes ago, courier brought me a package, thank you for a bounty ! :)

B Jira Ticket zur XSS Lücke

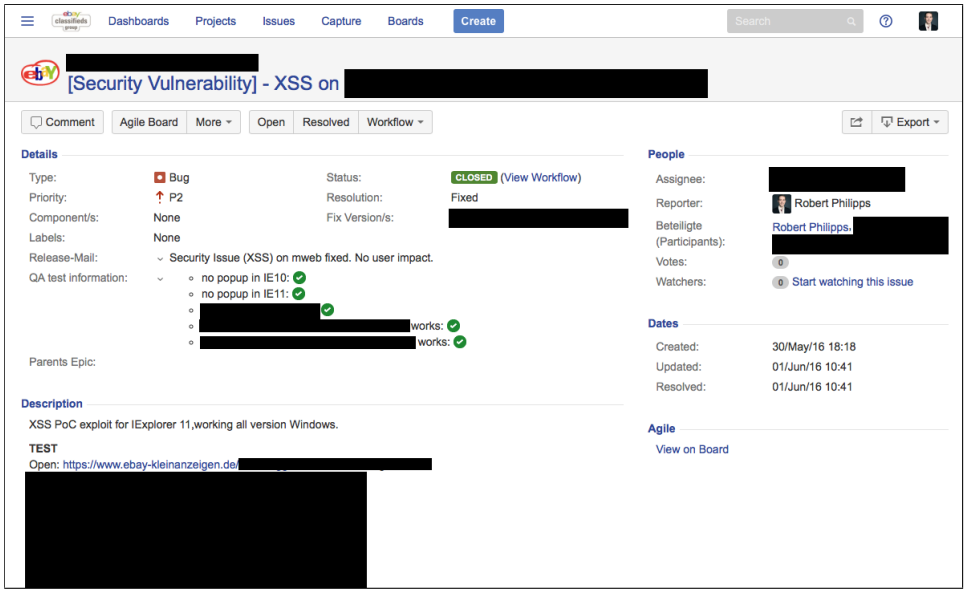


Abbildung 6: Das Jira Ticket zur genannten XSS Lücke.

C Google - Reihe von Fragen

C.1 Negatives Resultat

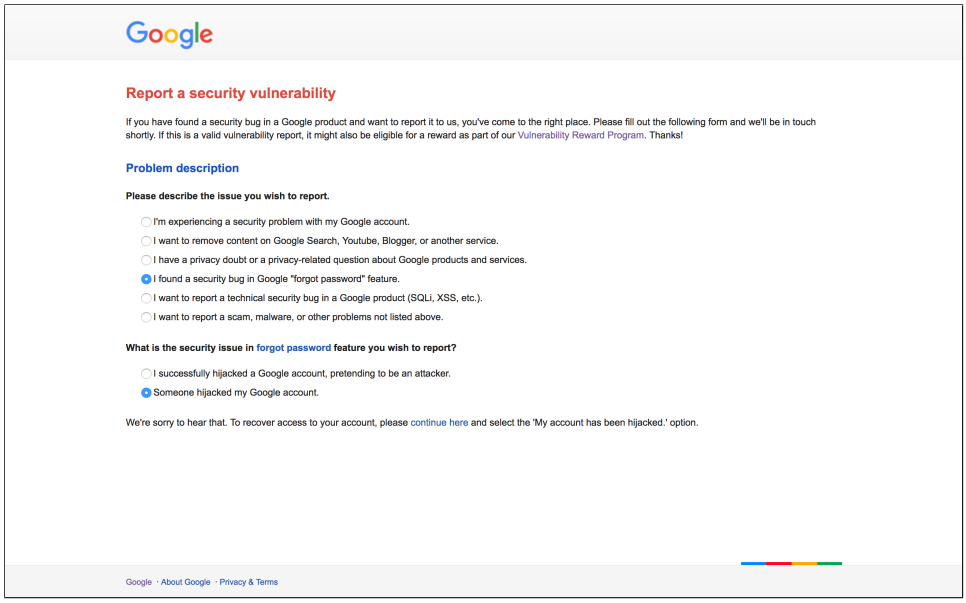


Abbildung 7: Das Webformular erkennt, dass es sich nicht um eine Sicherheitslücke, sondern ein Problem mit dem eigenen Nutzeraccount handelt.

C.2 Positives Resultat

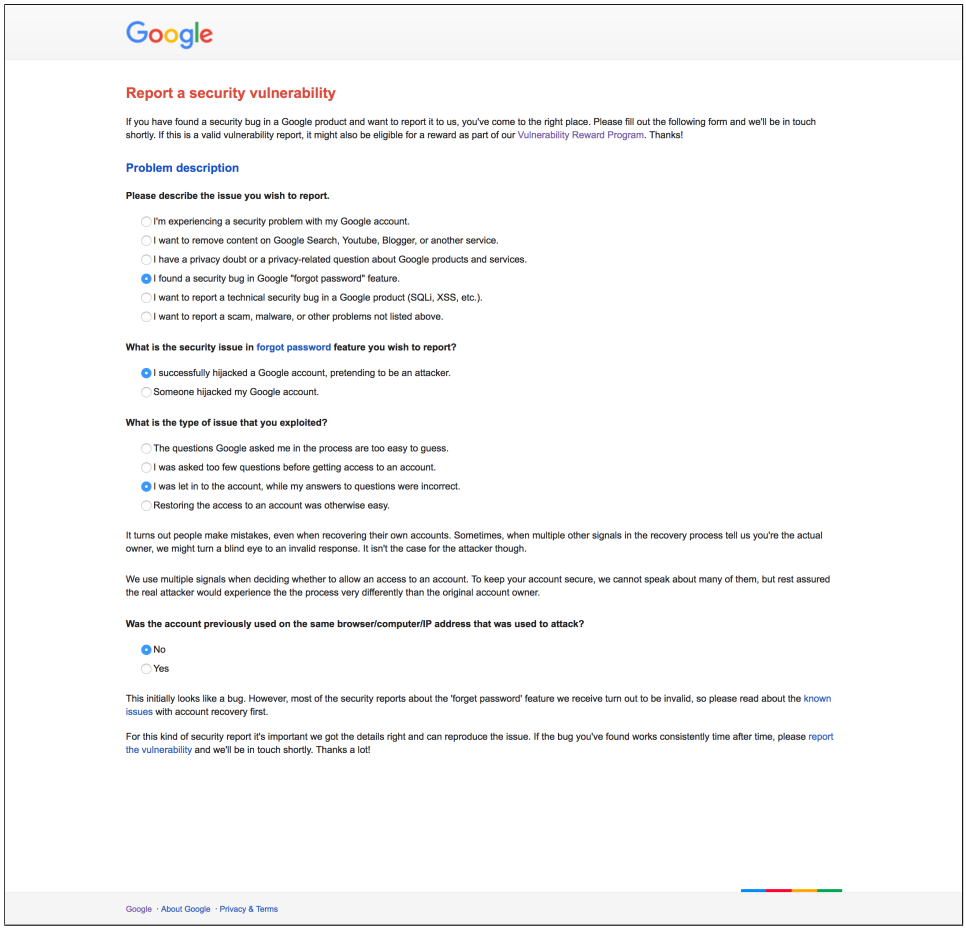


Abbildung 8: Das Webformular erkennt, dass es sich um eine Sicherheitslücke handeln könnte.

D Endorsment auf openbugbounty.org

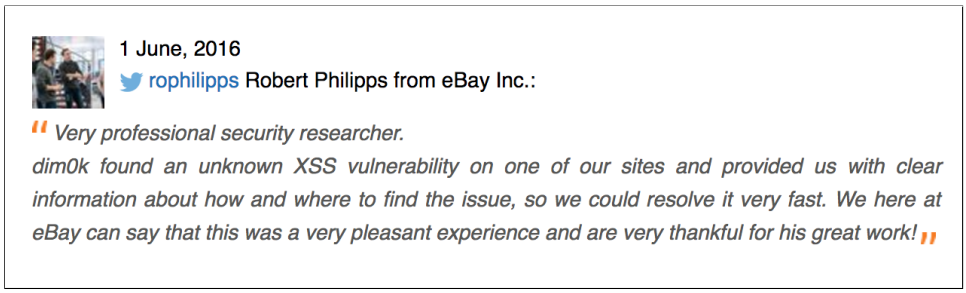


Abbildung 9: Das Endorsment auf der Profil Seite des Hackers. Zu finden auf openbugbounty.org/researchers/dim0k/.

Literatur

Alle elektronischen Quellen wurden zuletzt am 24. April 2017 um ca. 12 Uhr CEST abgerufen.

[1] T. Johnson, "Can you hack? here's a way to make a buck – and it's legal," Dec. 2016. [Online]. Available: http://www.bellinghamherald.com/news/nation-world/national/article123160729.html

[2] C. Bucholtz, "Netscape answers security lapses with anti-bug contest," *Network Security*, vol. 1995, no. 10, pp. 7–8, Oct. 1995.

[3] Alexa Internet (www.alexa.com), Apr. 2017. [Online]. Available: http://www.alexa.com/topsites/countries/DE

[4] A.-C. Hardy, "Agile team organisation: Squads, chapters, tribes and guilds," Feb. 2016. [Online]. Available: http://www.full-stackagile.com/2016/02/14/team-organisation-squads-chapters-tribes-and-guilds/

[5] G. Walker, *IT Problem Management*, 1st ed. Prentice Hall, Mar. 2001.

[6] M. Zhao, A. Laszka, T. Maillart, and J. Grossklags, "Crowdsourced security vulnerability discovery: Modeling and organizing bug-bounty programs," HCOMP Workshop on Mathematical Foundations of Human Computation, Nov. 2016.

[7] A. Austin and L. Williams, "One technique is not enough: A comparison of vulnerability discovery techniques," in *2011 International Symposium on Empirical Software Engineering and Measurement*. Institute of Electrical and Electronics Engineers (IEEE), Sep. 2011.

[8] H. Tuttle, "The 2017 cyberrisk landscape," *Risk Management*, vol. 64, no. 1, p. 4, 2017.

[9] M. Sheetz, *Computer Forensics: An Essential Guide for Accountants, Lawyers, and Managers*. Wiley, 2015. [Online]. Available: https://books.google.de/books?id=PCPPBwAAQBAJ

[10] Magento, "Magento bug bounty program," Aug. 2016. [Online]. Available: https://bugcrowd.com/magento

[11] Mastercard, "Mastercard bug bounty program." [Online]. Available: https://bugcrowd.com/mastercard

[12] Curesec Research Team, "Cve-2013-6271: Security advisory," Nov. 2013. [Online]. Available: https://www.curesec.com/data/advisories/Curesec-2013-1011.pdf

[13] GitHubSecurity, "Pgp key for bounty submissions." [Online]. Available: https://bounty.github.com/pgp.html

[14] Symantec, "Vulnerability management commitment and disclosure policy." [Online]. Available: https://www.symantec.com/security-center/vulnerability-management

[15] Dell, "Report a potential vulnerability in dell products or software." [Online]. Available: http://www.dell.com/learn/us/en/04/campaigns/report-vulnerability

[16] Google, "Report a security vulnerability." [Online]. Available: https://www.google.com/appserve/security-bugs/m2/new?rl=&key=

[17] Twitter, "Twitter: Bug bounty program," Jan. 2017. [Online]. Available: https://hackerone.com/twitter

[18] PayPal, "Paypal bug bounty program," 2017. [Online]. Available: https://www.paypal.com/us/webapps/mpp/security-tools/reporting-security-issues

[19] Google Security Team, "Google bughunter university: Improve your reports." [Online]. Available: https://sites.google.com/site/bughunteruniversity/improve

[20] T. Ring, "Why bug hunters are coming in from the wild," *Computer Fraud & Security*, vol. 2014, no. 2, pp. 16–20, Feb. 2014.

[21] S. Krishnamurthy and A. K. Tripathi, "Bounty programs in free/libre/open source software," in *The Economics of Open Source Software Development*. Elsevier BV, 2006, pp. 165–183.

[22] Google, "Chrome reward program rules." [Online]. Available: https://www.google.com/about/appsecurity/chrome-rewards/index.html

[23] ——, "Google security reward programs." [Online]. Available: https://www.google.com/about/appsecurity/programs-home/

[24] Microsoft Security Response Center, "Microsoft bounty programs." [Online]. Available: https://technet.microsoft.com/en-US/security/dn425036

[25] HackerOne, "Qualcomm launches bug bounty program," Nov. 2016. [Online]. Available: https://www.hackerone.com/blog/Qualcomm-launches-bug-bounty-program

[26] Tesla, "Responsible reporting process." [Online]. Available: https://bugcrowd.com/tesla

[27] Secretary of Defense, "Building on the lessons learned from hacking the pentagon," Nov. 2016. [Online]. Available: https://medium.com/@SecDef/building-on-the-lessons-learned-from-hacking-the-pentagon-b5b58548b6a9

[28] S. Collins, "Dod announces 'hack the pentagon' follow-up initiative," Oct. 2016. [Online]. Available: https://www.defense.gov/News/Article/Article/981160/dod-announces-hack-the-pentagon-follow-up-initiative

[29] Yahoo, "Yahoo bug bounty case study," May 2016. [Online]. Available: https://www.hackerone.com/resources/yahoo-bug-bounty-case-study

[30] N. Perlroth, "Hackerone connects hackers with companies, and hopes for a win-win," Jun. 2015. [Online]. Available: https://www.nytimes.com/2015/06/08/technology/hackerone-connects-hackers-with-companies-and-hopes-for-a-win-win.html?_r=0

[31] HackerOne, "What is hackerone's bug bounty service fee?" [Online]. Available: https://support.hackerone.com/hc/en-us/articles/204951979-What-is-HackerOne-s-bug-bounty-service-fee-

[32] H. Dalziel, "US cybersecurity jobs," in *How to Become an Ethical Hacker and Penetration Tester*. Elsevier BV, 2015, pp. 1–8.

[33] K. R. Lakhani and R. Wolf, "Why hackers do what they do: Understanding motivation and effort in free/open source software projects." *Perspectives on Free and Open Source Software*, pp. 1–27, 2005.

[34] N. Perlroth and D. E. Sanger, "Nations buying as hackers sell flaws in computer code," Jul. 2013. [Online]. Available: http://www.nytimes.com/2013/07/14/world/europe/nations-buying-as-hackers-sell-computer-flaws.html

[35] NTIA Awareness & Adoption Group, "Vulnerability disclosure attitudes and actions," 2016. [Online]. Available: https://www.ntia.doc.gov/files/ntia/publications/2016_ntia_a_a_vulnerability_disclosure_insights_report.pdf

[36] J. Tyson, "Facebook bug bounty: $5 million paid in 5 years," Oct. 2016. [Online]. Available: https://www.facebook.com/notes/facebook-bug-bounty/facebook-bug-bounty-5-million-paid-in-5-years/1419385021409053

[37] HackerOne, "2016 bug bounty hacker report," Sep. 2016. [Online]. Available: https://www.hackerone.com/blog/bug-bounty-hacker-report-2016

[38] Bugcrowd, "Inside the mind of a hacker," Feb. 2016. [Online]. Available: https://pages.bugcrowd.com/inside-the-mind-of-a-hacker-2016

[39] J. Lerner and J. Tirole, "Some simple economics of open source," *The Journal of Industrial Economics*, vol. 50, no. 2, pp. 197–234, Mar. 2003.

[40] J. Materna, "Why bounty programs are a good fit for the agile development process," Mar. 2017. [Online]. Available: https://blog.assembla.com/why-bounty-progams-are-a-good-fit-for-the-agile-development-process

[41] M. Finifter, D. Akhawe, and D. Wagner, "An empirical study of vulnerability rewards programs," in *Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13)*. Washington, D.C.: USENIX, 2013, pp. 273–288. [Online]. Available: https://www.usenix.org/conference/usenixsecurity13/technical-sessions/presentation/finifter

[42] "Bidding war for bug hunter rewards," *Network Security*, vol. 2005, no. 8, p. 3, Aug. 2005.

[43] The Economist, "How to manage the computer-security threat," Apr. 2017. [Online]. Available: http://www.economist.com/news/leaders/21720279-incentives-software-firms-take-security-seriously-are-too-weak-how-manage

[44] M. Staples, "Building a world-class bug bounty program (part i: An intro)," Feb. 2017. [Online]. Available: https://engineering.zenefits.com/2017/02/building-a-world-class-bug-bounty-program-part-i-an-intro/

[45] D. Brush, "Theresa payton: Design for the human," Mar. 2017. [Online]. Available: https://cybersecurityinterviews.com/019-theresa-payton-design-human/

[46] KasperskyLab, "Hacking america: Cybersecurity perception," Feb. 2017. [Online]. Available: https://usblog.kaspersky.com/daily-us/files/2017/02/RSA-Survey-Report-FINAL123.pdf

[47] K. Conger, "Yelp's bug bounty improves security and attracts talent," Feb. 2017. [Online]. Available: https://techcrunch.com/2017/02/13/yelps-bug-bounty-improves-security-and-attracts-talent/

[48] A. Carman, "Shellshock used to amass botnet and execute phishing campaign," Oct. 2014. [Online]. Available: https://www.scmagazine.com/bash-bug-used-to-assemble-botnet/article/540027/

[49] D. E. Denning, "Toward more secure software," *Communications of the ACM*, vol. 58, no. 4, pp. 24–26, Mar. 2015.

[50] Uber, "Uber: Bug bounty program," Mar. 2017. [Online]. Available: https://hackerone.com/uber

[51] Dropbox, "Dropbox: Bug bounty program," Jan. 2017. [Online]. Available: https://hackerone.com/dropbox

[52] A. Tetelman, "Bug bounty, 2 years in," May 2016. [Online]. Available: https://blog.twitter.com/2016/bug-bounty-2-years-in

[53] C. Osborne, "Yahoo changes bug bounty policy following 't-shirt gate'," ZDNet, Oct. 2013. [Online]. Available: http://www.zdnet.com/article/yahoo-changes-bug-bounty-policy-following-t-shirt-gate/

[54] R. Fletcher and L. Glovin, "Celebrating a year of smashing bugs," Mar. 2017. [Online]. Available: https://medium.com/uber-security-privacy/uber-bug-bounty-year-one-e0464bcfddd7#.9gdaqcoh5

[55] M. Feldman, "Slack bug bounty: Three years later," Mar. 2017. [Online]. Available: https://slack.engineering/slack-bug-bounty-three-years-later-ad59e9188603

[56] P. Mell, K. Scarfone, and S. Romanosky, "A complete guide to the common vulnerability scoring system," Jun. 2017. [Online]. Available: https://www.first.org/cvss/cvss-v2-guide.pdf

[57] HackerOne, "Intel launches its first bug bounty program," Mar. 2017. [Online]. Available: https://www.hackerone.com/blog/Intel-launches-its-first-bug-bounty-program

[58] M. Walton, "Nintendo offers $20,000 bounty for 3ds exploits," Dec. 2016. [Online]. Available: https://arstechnica.com/gaming/2016/12/nintendo-3ds-exploit-bounty/