**Technische Berichte in Digitaler Forensik**

**Herausgegeben vom Lehrstuhl für Informatik 1 der Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) in Kooperation mit dem Masterstudiengang Digitale Forensik (Hochschule Albstadt-Sigmaringen, FAU, Universität des Saarlandes)**

# PayPal's Bug Bounty Program and a Security Issue in it's Express Checkout API

Florian Hantke
florian.hantke@fau.de

01.07.2020

Technischer Bericht Nr. 19

**Zusammenfassung**

PayPal ist eine amerikanische Firma, die nutzerfreundliche weltweite online Zahlungen ermöglicht. Für eine Firma dieser Größe können Fehler (Bug) im Zahlungssystem und der daraus resultierende Vertrauensverlust schwere Konsequenzen haben. Auch für forensische Analysen ist das Wissen um solche Bugs und Schwachstellen essentiell, beispielsweise bei einem Fall in dem es um Zahlungsbetrug geht. Nichtsdestotrotz gibt es wiederholt Beschwerden, dass PayPal den Umgang mit solchen Fehler nicht ernst nimmt. In diesem Bericht geben wir einen Überblick über die Geschichte des PayPal But Bounty Programms und seine Probleme. Im Zuge dessen präsentieren wir eine Sicherheitslücke die wir gefunden haben und betrügerische Abrechnungen ermöglicht, bei PayPals Bug Bounty Programm aber nicht akzeptiert wurde. Der Bericht endet mit Vorschlägen, wie PayPal und auch andere Firmen ihre Bug Bounty Programme verbessern könnten.

Entstanden im Rahmen des CTF Teams FAUST am Lehrstuhl für Informatik 1 der Friedrich-Alexander-Universität unter der Anleitung von Felix Freiling

# PayPal's Bug Bounty Program and a Security Issue in it's Express Checkout API

Florian Hantke
florian.hantke@fau.de

01.07.2020

## 1 Introduction and Motivation

PayPal is an American company that enables user-friendly online payments all around the world. According to their annual global impact report, in 2019, PayPal had 305 million active users and handled 12.4 billion transactions [1]. For such a large company, bugs in their system and the resulting loss of user confidence can lead to devastating consequences, as can be seen with the company Equifax [2]. Furthermore, for forensic investigations the understanding of potential security issues caused by bugs is important, for instance in case of payment fraud. Nevertheless, there are many complaints that PayPal does not take the threat of potential bugs seriously, even after they started their bug bounty program on HackerOne [3].
This report starts with a short overview of the history of PayPal's bug bounty program and problems. Afterwards, we present an issue that we disclosed to PayPal that was not accepted, a bug that can easily be abused to allow excess fraudulent charges where the buyers are unknowingly charged more than they agreed to pay. In the end, we propose aspects that we think would improve not only PayPal's program further, but also guide other companies in developing their own program.

## 2 History of PayPal's Bug Bounty Program

The first-ever bug bounty program was launched 1995 by Netcap [4]. Since then, the number of active bug bounty programs has increased tremendously [5]. In 2012 PayPal opened its own bug bounty program and rewarded roughly 2000 ethical hackers over the next seven years [6]. However, there was a notable amount of criticism that PayPal does not communicate transparently with the researchers [7, 8]. For instance, PayPal refused to reward a 17-year old student who found a bug, as he was too young to qualify for their program.
In 2018 PayPal decided to move its program to HackerOne and started its official managed program with a maximum award of 10000$ [9]. With the success of this program, the company decided to expand its bounty further to a maximum of 30000$ in the following year [10]. Nevertheless, researchers still note various problems with PayPal's program in transparency and its courtesy. In February 2020, a research team from Cybernews reported six individual bugs to PayPal via HackerOne [3]. None of these reports were rewarded as some were duplicates or out of scope. The issue that we present in the next section was also classified as out of scope. As outlined in the preceding examples, the scope is commonly misunderstand, thus we believe that PayPal should consider changes in their program scope which is proposed in the discussion.

# 3 Security Issue in the PayPal Express Checkout API

PayPal offers various APIs giving developers the option to integrate PayPal payments in their homepages and workflow [11]. The security flaw in this article uses PayPals *Express Checkout (NVP/SOAP)*, an API to integrate PayPal checkouts into online shops. Although the API is deprecated as of January 1, 2017, we found it still widely used by numerous web stores.

## 3.1 Attack Overview

Figure 1 shows the workflow of the API [12]. In the first step, the buyer decides to use Paypal as their checkout method. Subsequently, the back-end sends a request to PayPal where details about the purchase are shared, such as the price. PayPal then sends a token back to the store that is used to redirect the buyer to the PayPal website. Once on PayPal's official website, the user must log in, review the bill and agree to the purchase. After the purchase is confirmed, PayPal redirects the user back to the store where the back-end then uses the token and payer-id to approve the transaction.

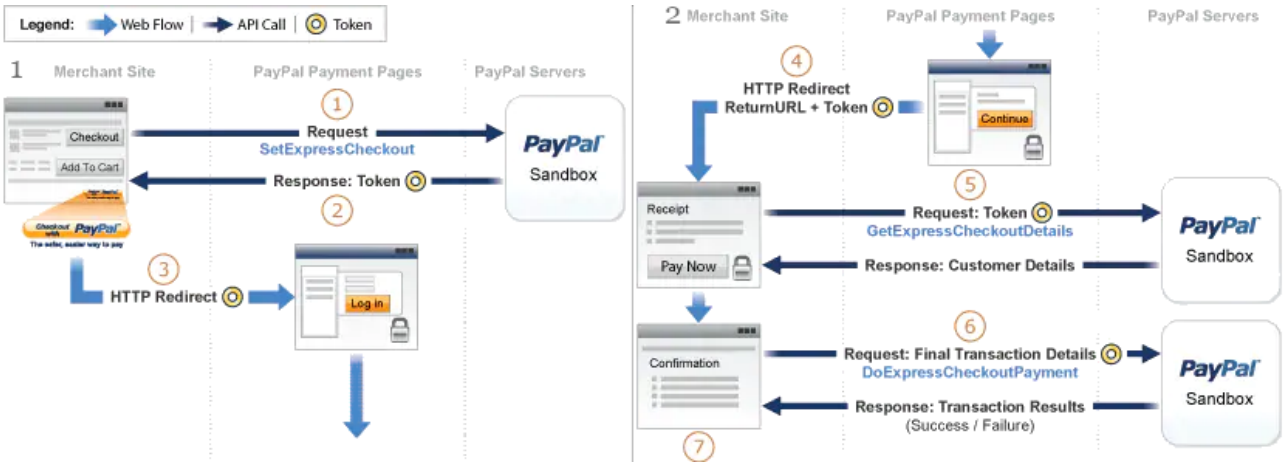


Figure 1: Paypals Express Checkout API workflow [13]

This is a solid workflow to ensure that the buyer cannot modify the price. However, we discovered that the seller is able to manipulate the price after the buyer approval step on the PayPal website (Step 4). During the last step (Step 6), the store server sends a final confirmation back to PayPal but the seller can arbitrarily change the price. PayPal does not check whether the price is the same as the price the buyer agreed to. In our tests, PayPal indeed charged the new/modified price without the knowledge of the buyer. Buyers can find the true charge amount in their PayPal account summary, but, we believe that only a few people actually check their account summary regularly.

## 3.2 Impact

A malicious seller could fool customers into paying more than they agreed to. The seller could, for instance, create a web store that charges double the price the customer accepted. If the customers do not check their account summary after the purchase and only trust the price they agreed to on the official PayPal webpage, they could potentially be overcharged. Another more obscure method is to add a small additional charge, such as 50 cents, to the price after the buyer confirms the transaction on PayPal. Even if the buyer realizes that the charge is not the same they agreed to, they may mistakenly rationalize it as a surcharge for PayPal's fees and

services. This can negatively affect PayPal's reputation or worse, make PayPal compensate the victims of this bug.

### 3.3 Disclosure

We disclosed this behavior to PayPal via HackerOne on the 23rd of April, 2020. PayPal did not accept this bug as it doesn't seem to be a security risk and/or have a security impact. After further inquiries, they told us the bug does not fit their bug bounty policy, since attacks involving payment fraud, theft, or malicious merchant accounts are out of scope. We accept their decision, nevertheless, we truly believe in the transparency of companies and that customers should be informed about questionable conduct. Under these circumstances, we decided to publish this report in the hope that PayPal will fix this issue and rethink its disclosure policy.

## 4 Discussion

We have seen that PayPal was often criticized for not being cooperative with security researchers in the past. Still, after moving to HackerOne researches disagree with PayPal's policy. In the following we discuss this criticism and propose some modifications to the policy.
Setting a clear scope is important for a company, however the scope should be well thought and understandable. In PayPal's bug bounty policy, it is written that: "Attacks involving payment fraud, theft, or malicious merchant accounts" [9] are out of scope. While we understand that a malicious merchant is not a bug of a system, we would argue that a bug that enables fraud should be in scope of the program. PayPal cannot prevent a shop not delivering on the promised product, yet PayPal can prevent a merchant from changing the price after the customers' agreement.
Furthermore, two of the vulnerabilities that the research team at Cybernews found were out of scope as they require already stolen credentials. With these, they were able to bypass two factor authentication and other behavior-based fraud detection. Even so credentials by itself should be secure, though they should not be an exclusion criterion for the bug bounty program. Multiple incidences have shown that leaked credentials exist and PayPal should also take security mechanisms besides credentials seriously.
With both examples in mind, PayPal makes people think that with their bug bounty program they only secure their infrastructure, yet the security of the user is left behind. Although, the aforementioned commentary presents critique, PayPal does many things correctly in its program and we want to point them out as well.
PayPal learned from previous mistakes and lowered the age limit. It states explicitly that attendees are required to be older than 14. Additionally, PayPal gives clear rules what they expect to be part of a report. That is very helpful and guiding creating a report. Another positive fact is that the scope in terms of domain addresses is large and includes wild cards. This means, with "*.paypal.com" they not only accept "paypal.com", but also all sub-domains. We saw many programs restricted to only a few sub-domains, however a real world hacker would not be bounded by the restriction either. Lastly, PayPal offers a great sandbox environment in which developers can test payment workflows. This is great as security researchers can also test payloads with little risk of real damage. Other companies even offer test accounts to check premium services without a cost to the researcher.
In summary, we propose that companies should set a clear scope that includes all their infrastructure and only excludes critical areas. Within the scope, not only the security for the company's infrastructure, but also for the user should be considered. Finally, a well designed testing environment or special test accounts should be provided.

# 5 Conclusion

In this report, we gave a short overview on PayPal's bug bounty program and discussed existing criticism. Furthermore, we presented our own experience and discussed the security issue we found in PayPal's Express Checkout API. In the discussion section, we provided points to consider which can help companies to design or improve their bug bounty program.

# References


[1] PayPal. April 2020. Global Impact Report. PayPal. https://investor.paypal-corp.com/static-files/4591acef-6f93-444f-81ae-6f1c0636e2e6 visited May 23, 2020.

[2] Lily Hay Newman. September 2017. Equifax Officially Has No Excuse. Wired. https://www.wired.com/story/equifax-breach-no-excuse visited May 23, 2020.

[3] Bernard Meyer. February 2020. We found 6 critical PayPal vulnerabilities – and PayPal punished us for it. Cybernews. https://cybernews.com/security/we-found-6-critical-paypal-vulnerabilities-and-paypal-punished-us visited May 23, 2020.

[4] Netscape. May 1997. "Netscape Bugs Bounty" with Release of Netscape Navigator 2.0 Beta. Netscape http://web.archive.org/web/19970501041756/www101.netscape.com/newsref/pr/newsrelease48.html visited May 23, 2020.

[5] Thomas Maillart, Mingyi Zhao, Jens Grossklags, John Chuang. June 2017. Given enough eyeballs, all bugs are shallow? Revisiting Eric Raymond with bug bounty programs. Journal of Cybersecurity, Volume 3, Issue 2. https://academic.oup.com/cybersecurity/article/3/2/81/4524054 visited May 23, 2020.

[6] HackerOne. September 2019. PayPal Celebrates its first Anniversary on HackerOne. HackerOne. https://www.hackerone.com/blog/paypal-celebrates-its-first-anniversary-hackerone visited May 23, 2020.

[7] Jeremy Kirk. May 2013. PayPal denies teenager reward for finding website bug. ITWorld. https://www.itworld.com/article/2828880/paypal-denies-teenager-reward-for-finding-website-bug.html visited May 23, 2020.

[8] l8security. October 2012. PayPal Bug Bounty - a lesson in not being a fuckup. https://web.archive.org/web/20121021222451/http://l8security.com/post/33876600904/paypal-bug-bounty-a-lesson-in-not-being-a-fuckup visited May 23, 2020.

[9] PayPal. July 2019. Bug Bounty Program. HackerOne. https://hackerone.com/paypal visited May 23, 2020.

[10] Jessica Haworth. March 2018. PayPal bug bounty increases to $30k. The Daily Swig. https://portswigger.net/daily-swig/paypal-bug-bounty-increases-to-30k visited May 23, 2020.

[11] Docs Archive. PayPal. https://developer.paypal.com/docs/archive visited May 23, 2020.

[12] Express Checkout NVP/SOAP (Deprecated) - NVP/SOAP Integration. PayPal. https://developer.paypal.com/docs/archive/express-checkout visited May 23, 2020.

[13] Express Checkout NVP/SOAP (Deprecated) - Test Your Integration. PayPal. https://developer.paypal.com/docs/archive/express-checkout/ec-test-your-integration visited May 23, 2020.