Sam Yarcho

SE-4930 Secure Software

Article Summary 2

1/10/12

How to Shop for Free Online

IEEE Transaction on Security Journal Article

This article explains some of the security problems many multiparty shopping websites face, how to exploit them, and how the exploits can be fixed by the exploited parties. Many websites are starting to use third-party services to pay for their product these days. These services are called Cashier-as-a-Service, or CaaS. Examples of these are PayPal, Amazon Payments, and Google Checkout. Integrating these services into the merchant website introduces new security challenges and opens up many new exploits that are not there in single party shopping websites that require no outside communication with other parties. A malicious shopper can use these exploits to shop for free once one item is purchased, get prices very low, or even shop completely for free. Most of these exploits are logic flaws that are caused by improper distribution of the application functionality between the client and the server (happening when the server relies on the client logic to validate user privileges, for instance). An example found was a merchant using the Amazon Payments service. The shopper's browser talks to the merchant server to place an order, which talks to the Amazon server to make a payment. If the payment is made to the shoppers own Amazon seller account, the merchant server will think it is fully paid, and allow the shopper to shop for free.

The funny thing is, a study showed that 59% of US online shoppers are more likely to use a website that accepts CaaS payment methods (multi-party payment methods) rather than a payment method built into the site (single party website), because these payment methods are trusted. But really, why wouldn't someone trust PayPal, Amazon payments, or Google checkout? A single party website could be a scam too, as they are not big names like PayPal, Amazon, and Google. The authors of this article did some investigating and tried exploiting some well known merchants using third-party CaaS payment methods. They show six pictures of items they got for free by exploiting the system, all obtained legally as it was closely advised by their lawyer. They found a security flaw in the Amazon Payments SDK that caused Amazon to significantly alter the way for verifying its payment notifications. They then reported all their findings to the merchants they bought from, and were thanked, and the problems were fixed or put on high priority.

To understand the nature of the security threats that websites using CaaS payment systems are facing, the authors explain that we must first understand the security goals of the system and the challenges achieving it. According to the article, below are online shopping websites security goals.

Say Merchant M changes the status of an item I to "paid" with regard to a purchase being made by Shopper S. This can only happen if:

1. M owns I
2. A payment is guaranteed to be transferred from an account of S to that of M in the CaaS.
3. The payment is for the purchase of I, and is valid for only one piece of I.
4. The amount of this payment is equal to the price of I.

This is what they call their security invariant. Preserving this invariant is very complex, and is what leads to several problems that can be used as exploits. One example of a problem is that the merchant might not understand how the CaaS system works completely, and take a confirmation of payment notification from CaaS to mean that the payment was made, and believe it, instead of checking to make sure which is what the CaaS may say the merchant must do. The article goes into many more technical exploits that can be used in this; however they are out of the scope of this summary.

The authors then start going over many popular CaaS systems and how they can be exploited. Mostly through the use of passing arguments in the HTTP requests, arguments which they get from looking at source code they show you how to find. It also shows you how to exploit CaaS systems that are closed source by using arguments as well, gained from looking at other HTTP requests and getting the argument names from there. The authors then cover attacker anonymity, and show how a malicious user can stay anonymous. They bought prepaid credit card gift cards, and registered them under fake names. Then they created Amazon/Google/PayPal accounts under fake names and emails using Gmail. Some of the things they bought for free with exploits were digital, which required no shipping. For physical shipping, attackers only need to have the item delivered to an address unlinked to them and sign for delivery using their fake name. The article then goes on to say all the information they gave to the websites they exploited to help those companies avoid this in the future. They list all the different merchants and CaaS systems they exploited, and detailed what they explained to them in order to help them. They told them how to prevent attacks, how to detect attacks, and even went as far as writing C programs for that will prevent attacks like the ones they showed. They then concluded the article by summarizing everything they presented, and saying that their study "takes the first step in the new security problem space that hybrid web applications bring to us.", and admit they have only scratched the surface.

I think this article is very interesting in that it shows in very great detail how they were able to exploit multiple merchants in many different ways, and how they were able to help the merchants fix it. It really helps that I took Web Applications with Dr. Hornick last term because there was a lot of HTTP process talk in this article. There were many diagrams as well that made this article easier to understand. I'm sure there are still websites out there that have not fixed the exploits explained in this article, and hopefully they hear of these exploits soon to avoid future theft. I learned a lot about how multiparty shopping websites work, how they try to prevent exploits, and how they can be exploited.