

PLAYING IN A SANDBOX WITH AUTHENTICATED SECURITY

(700100)

*700100: REPORT*

*TRUSTWORTHY COMPUTING*

*18 MAY 2020*

Farah Aly - 201505395

Contents

[Introduction 1](#_Toc40548788)

[Report 1](#_Toc40548789)

[API 1](#_Toc40548790)

[Definition 1](#_Toc40548791)

[How an API works 1](#_Toc40548792)

[API response 1](#_Toc40548793)

[Inside the PayPal Sandbox and REST API 1](#_Toc40548794)

[Weakness 2](#_Toc40548795)

[Dummy Catering System 2](#_Toc40548796)

[PayPal Sandbox 2](#_Toc40548797)

[Short Note 4](#_Toc40548798)

[Authenticated Encryption Modes 4](#_Toc40548799)

[Generic Composition Methods of Authenticated Encryption Modes 4](#_Toc40548800)

[Single-Pass Authenticated Encryption Modes 4](#_Toc40548801)

[Conclusion 5](#_Toc40548802)

[References 6](#_Toc40548803)

[Appendix 7](#_Toc40548804)

[Appendix A: Pseudo Code 7](#_Toc40548805)

[Appendix B: Tests 8](#_Toc40548806)

[Appendix B: Risks 9](#_Toc40548807)

# Introduction

This technical document is divided into two parts. It describes APIs. What information they take, how they work and how the PayPal REST API is managed and explains Authenticated Encryption Modes.

The first part is the report which is divided into 3 parts. It will go in depth into APIs, the PayPal Sandbox and the weaknesses in the system. The second part is the short notes which will be depicting the concept of authenticated encryptions modes, their generic composition methods and single-pass authenticated encryption modes. The report then has a conclusion, reference and appendices which contain the pseudo code, the tests and the results of the heuristic evaluation.

# Report

## API

### Definition

API stands for Application Programing Interface. It is an intermediate software that allows communication between two applications (Pearlman, 2016). An API is a set of protocols that let the main software applications to be used by another application safely. APIs can be thought of as public methods of an object-oriented program that interact with other elements on the application.

There are different types of API. APIs are divided into four categories: public APIs, partner APIs, internal APIs and composite APIs. Public or Open APIs are APIs that are made publicly available by the software developers. There are to be used free meaning that the software owners give universal access to their customers to integrate this API into their own systems. For example, Facebook’s API allows third-party tools to post on their user’s feed. Partner APIs are not publicly available. Developers will need specific rigts or licenses in order to be able to implement them.

* API Architect. REST vs SOAP

### How an API works

### API response

## Inside the PayPal Sandbox and REST API

* Where encryption takes place

**HOW DO YOU SET UP AN ACCOUNT**

* Step 1: Go to PayPal Developer.
* Step 2: A personal and business accounts are made by default. The personal account has been renamed to 'buyer@uoh.ac.uk'; the business account has been renamed to 'seller@uoh.ac.uk'
* Step 3: Go to 'My Apps & Credentials' and 'Create App' for this project: University\_Of\_Hull\_Catering\_Srvc:
  + The sandbox account is set as: seller@uoh.ac.uk
  + The client ID is:

AYbCNBxQmjqgAFsJIjQpMI30072iJoU5xDSMYrPx9SxM12zjD14\_93\_9xuVbJCV-8FcfBGFIEztLEZMY

* This authenticates API requests from this account. These credentials are used when testing API calls to the sandbox environment. This credentials come from a REST API in the developer dashboard.

* What credentials issued by paypal

**PAYPAL API, HOW TO SET UP AN ACCOUNT, HOW TRANSACTION WORKS**

* To accept payments, retailers must integrate a set of payment calls (for capturing, authorizing, settling payments, etc.) that, all together, enable users to complete a transaction.

To include payments in your commerce you can either use API from some provider or use a URL redirect offsite gateway where all payment interface is done on some 3rd party service provider

From <<https://www.quora.com/Laymans-Explanations-What-is-a-payments-API>>

* When the PayPal button is clicked, the PayPal API is called to set up the payment. The checkout flow is then started in the browser.
* PayPal used REST API: Representational State Transfer -  is designed to take advantage of existing protocols. While REST can be used over nearly any protocol, it usually takes advantage of HTTP when used for Web APIs. This means that developers do not need to install libraries or additional software in order to take advantage of a REST API design.

There are six key constraints to REST API design to be aware of when deciding whether this is the right [API type](https://www.mulesoft.com/resources/api/types-of-apis) for your project:

* Client-Server
* Stateless
* Cache
* Uniform Interface
* Layered System
* Code on Demand
* From <<https://www.mulesoft.com/resources/api/what-is-rest-api-design>>

## Weakness

### Dummy Catering System

* No sessions
* Login, for now hard coded, stored in localstorage
* If localstorage cleared, order is deleted
* After checkout, if the users goes back to previous page, the order can be processed again
* No way to save the orders previously made
* Order summary doesn’t show in paypal email

### PayPal Sandbox

* Glitches in accounts editing
* API time changes
* Where encryption takes place

**HOW DO YOU SET UP AN ACCOUNT**

* Step 1: Go to PayPal Developer.
* Step 2: A personal and business accounts are made by default. The personal account has been renamed to 'buyer@uoh.ac.uk'; the business account has been renamed to 'seller@uoh.ac.uk'
* Step 3: Go to 'My Apps & Credentials' and 'Create App' for this project: University\_Of\_Hull\_Catering\_Srvc:
  + The sandbox account is set as: seller@uoh.ac.uk
  + The client ID is:

AYbCNBxQmjqgAFsJIjQpMI30072iJoU5xDSMYrPx9SxM12zjD14\_93\_9xuVbJCV-8FcfBGFIEztLEZMY

* This authenticates API requests from this account. These credentials are used when testing API calls to the sandbox environment. This credentials come from a REST API in the developer dashboard.

* What credentials issued by paypal

**PAYPAL API, HOW TO SET UP AN ACCOUNT, HOW TRANSACTION WORKS**

* To accept payments, retailers must integrate a set of payment calls (for capturing, authorizing, settling payments, etc.) that, all together, enable users to complete a transaction.

To include payments in your commerce you can either use API from some provider or use a URL redirect offsite gateway where all payment interface is done on some 3rd party service provider

From <<https://www.quora.com/Laymans-Explanations-What-is-a-payments-API>>

* When the PayPal button is clicked, the PayPal API is called to set up the payment. The checkout flow is then started in the browser.
* PayPal used REST API: Representational State Transfer -  is designed to take advantage of existing protocols. While REST can be used over nearly any protocol, it usually takes advantage of HTTP when used for Web APIs. This means that developers do not need to install libraries or additional software in order to take advantage of a REST API design.

There are six key constraints to REST API design to be aware of when deciding whether this is the right [API type](https://www.mulesoft.com/resources/api/types-of-apis) for your project:

* Client-Server
* Stateless
* Cache
* Uniform Interface
* Layered System
* Code on Demand
* From <<https://www.mulesoft.com/resources/api/what-is-rest-api-design>>

# Short Note

## Authenticated Encryption Modes

## Generic Composition Methods of Authenticated Encryption Modes

## Single-Pass Authenticated Encryption Modes

# Conclusion

# References

Pearlman, S., 2016. *What Are Apis And How Do Apis Work*?. [online] MuleSoft Blog. Available at: https://blogs.mulesoft.com/biz/tech-ramblings-biz/what-are-apis-how-do-apis-work/ [Accessed 13 May 2020].

# Appendix

## Appendix A: Pseudo Code

## Appendix B: Tests

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PAGE** | **#** | **TEST CONDUCTED** | **EXPECTED RESULT** | **ACTUAL RESULT** |
| Home | 1 | Click login button | Get login pop up |  |
|  | 2 | Click login button with no details | Error shows up |  |
|  | 3 | Click login button with no user ID | Error show up |  |
|  | 4 | Click login button with no password | Error show up |  |
|  | 5 | Click log in with wrong password | Error show up |  |
|  | 6 | Click log in with non-existent account | Error show up |  |
|  | 7 | Login as admin | Redirected to admin |  |
|  | 8 | Login as buyer1 | Redirected to index |  |
|  | 9 | Login as buyer2 | Redirected to index |  |
|  | 10 | Logout button | Redirect to home |  |
|  | 11 | Go to Admin button | Redirect to admin |  |
|  | 12 | Continue Shopping button | Redirect to index |  |
|  |  |  |  |  |
| **Admin** | 13 | Switch between tabs | Show different admin options |  |
|  | 14 | Products – Click on ‘Change Details’ | Go to coming soon |  |
|  | 15 | Stock – Open accordion tab | Show relevant table |  |
|  | 16 | Stock – Open multiple accordion tabs | Show multiple tables |  |
|  | 17 | Stock – Close accordion tab | Close relevant table |  |
|  | 18 | Order – Switch through tabs | Show relevant table |  |
|  | 19 | Go back to login page | Tell the user they’re already logged in |  |
|  | 20 | Logout button on nav bar | Go to home |  |
|  |  |  |  |  |
| **Index** | 21 | Switch between tabs | Show relevant pages |  |
|  | 22 | Click on Home | Go to home |  |
|  | 23 | Click on About | Go to about |  |
|  | 24 | Click on Contact | Go to contact |  |
|  | 25 | Click on cart | Go to cart |  |
|  | 26 | Hover on item | Add to Cart popup |  |
|  | 27 | Add item to cart | Cart icon + 1 |  |
|  | 28 | Add same item to cart | Increment in local storage |  |
|  | 29 | Add multiple items to cart | Add in local storage |  |
| **ShoppingCart** | 30 | Go back to login page | Tell the user they’re already logged in |  |
|  | 31 | Logout button on nav bar | Go to home |  |
| **PAGE** | **#** | **TEST CONDUCTED** | **EXPECTED RESULT** | **ACTUAL RESULT** |
| **ShoppingCart** | 32 | Tests 22 - 25 | Go to relevant page |  |
|  | 33 | Click on Continue Shopping | Go to index |  |
|  | 34 | Click on Proceed to Payment | Go to payment |  |
|  | 35 | Click on logout | Go to home |  |
|  |  |  |  |  |
| **Payment** | 38 | Tests 22 -25 | Go to relevant page |  |
|  | 39 | Click on Pay by Cash | Go to cashCheckout |  |
|  | 40 | Click on Pay by Paypal | Get Paypal pop up |  |
|  | 41 | Click on logout | Go to home |  |
|  |  |  |  |  |
| **CashCheckout Successful** | 42 | Make a new order | Go to index and clear cart |  |
|  | 43 | Click back button |  |  |
|  | 44 | Click on logout | Go to home |  |
|  |  |  |  |  |
| **PayPal payment pop-up** | 45 | Click on top-right drop down | See product details |  |
|  | 46 | Click proceed to payment | Go to PaypalCheckout |  |
| **PaypalCheckoutSuccessful** | 47 | Make a new order | Go to index and clear cart |  |
|  | 48 | Click back button |  |  |

## 

## Appendix C: Heuristic Evaluation Results