IOWA STATE UNIVERSITY

Pentest a Payment Provider

Final Report

Presented by

Senior Design Team sdmay21-06
Max Solaro
Ryan Anderson
Nathan Key
Matthew Maiman
Jacob Conn
Priyanka Kadaganchi
KayAnne Bryant

Client and Advisor: Dr. Benjamin Blakely

sdmay21-06@iastate.edu

TABLE OF CONTENTS

PROJECT DESIGN **3**
- Problem Statement 3
- Solution 3
- Scope 3
- Intended Users and Uses 4
- Deliverables 4

REQUIREMENTS **6**
- Rules of Engagement 6
- Standards 7

PROJECT PLAN **8**
- Timeline 8
- Testing Methodology 9
- Design Evolutions 11

IMPLEMENTATION **13**
- Client Information 13
- Staging 13

TESTING **14**
- Risk Ratings 14
- Testing Process 15
- Testing Results 16

APPENDICES **18**
- Appendix I -- Vulnerability Table 18
- Appendix II -- Project Alternatives 29
- Appendix III -- Other Considerations 29

PROJECT DESIGN

Problem Statement

The payment provider Dwolla is consulting with a team of cybersecurity professionals to perform a thorough penetration test against the Sandbox environment provided by Dwolla for their clients. The cybersecurity actors will perform as Red Team to assess Dwolla's API, Sandbox dashboard, and Sandbox account systems for known security vulnerabilities. Because the Sandbox environment is public-facing, discovered vulnerabilities must be validated and risk assessed. It is critical that Dwolla receive every security perspective possible to flush out any potential risk present on their service backbone.

Solution

Properly executing a penetration test requires cohesion between both actors and the client. A clearly defined Scope and Rules of Engagement must both be drafted before any intrusion into the problem area. These two items will be well stated further below in this documentation. Requirements and limitations keep the actors focused within the scope and prevent unintended breaches. During the test, the Red Team is expected to discover, validate, and appraise severity towards the security vulnerabilities found within the defined scope. The goal, therefore, is to identify these potential vulnerabilities and determine their exploitability. To accomplish this, an evaluation and exploitation methodology will be devised by Red Team as agreed upon with Dwolla. This methodology, coupled with the vulnerabilities and their appraised risk, will be compiled into two final deliverables. These deliverables include an Executive Summary and Technical Report. Potential remediation approaches will be discussed with Dwolla following completion of the penetration test.

Scope

We have been given a well-defined operational scope for testing purposes. This operational area is denoted simply as the "Sandbox environment". This environment, as opposed to the "Production environment", does not service live customers. Therefore, to prevent interfering with business operations, we are confined to testing within this area. Within this environment, we are assigned to test the following three domains:

https://api-sandbox.dwolla.com
https://accounts-sandbox.dwolla.com
https://dashboard-sandbox.dwolla.com
Sandbox API endpoints

This is the primary API interface that services Dwolla clients to communicate with the backend functionality/databases. This API mirrors the production API, and therefore security weaknesses found here will reflect issues of the production environment. For this reason, our findings are of a sensitive nature and may have to be redacted per Dwolla's request.

Intended Users and Uses

The intended audience for the project deliverables is Dwolla and their clients. Dwolla is the primary recipient and audience of the final report deliverables and will receive these documents in a clear, direct manner. Both technical and non-technical employees at Dwolla should be able to apply the information provided. Other users interested in our testing and findings may also receive or view these documents publicly with redacted information. It is the intention of this team to draft these documents in such a way that Dwolla may use them to their benefit in patching, remediation, and controls implementation to reduce security risk.

Deliverables

Scope and Rules of engagement

These documents will ensure that the penetration test against Dwolla is performed to the client's specifications. The scope will make certain, guarantee, certify that the penetration test is only against the systems designated by the client. The rules of engagement will outline how the penetration test is to be conducted, such as the type of tests performed.

Testing Methodology

After finalizing the scope and rules of engagement, the methodology for the penetration test will be developed around them. The methodology will lay out the methods and tools that will be used by the team. The methods outlined in the document will cover how the team will collect information, analyze vulnerabilities, proceed with exploitation, handle information post-exploitation, and report results. The tools outlined in the methodology will be centered around the constraints of the scope and rules of engagement. The tools used during the pentest will consist of open-source Linux software. Both the methods and tools will be based on the most common vulnerabilities affecting web applications and API software.

Executive Summary

The executive summary is one of the major documents of the final report to the client. This document will cover the results of the penetration test from a high level. This document will include details such as the purpose of the penetration

test, general findings, risk ranking for each of the vulnerabilities, and plans to remedy each of those vulnerabilities. This report will provide Dwolla's non-technical staff an insight into the security concerns from a business-oriented standpoint.

Technical Report

The technical report is the second major document for the final report to the client. This document will cover the result of the penetration test from a more technical level. This document will include details such as the vulnerability assessment, confirmation of the exploited vulnerabilities, techniques, and accessed systems post-exploitation, and the impact that the vulnerabilities had on the system. Dwolla's technical staff will use this report to understand vulnerabilities that were exploited in their systems and consider how they will remedy the issues based on vulnerability impact.

REQUIREMENTS

Rules of Engagement

Throughout the duration of the penetration test, the team must adhere to these strict rules of engagement laid out by our client Dwolla.

- Team members are authorized to and have been given permission to complete the pentest within the agreed-upon rules and scope and shall not be penalized in any way for carrying out any tasks associated with the test.
- All team members (7 members) will have access to the testing environment and have the ability to create an account.
- Team members will have access to all necessary equipment when needed.
- Team members should not be required to purchase equipment or software for the test.
- Team members will be available for scheduled meetings unless specified in advance.
- Team members will be allowed to ask questions and inquire about network and software details.
- If the team can locate no vulnerabilities, a separate, vulnerable system will be set up to allow the team to complete the test.
- The requirements for successfully completing this project within the scope of Iowa State will be malleable and adapted to fit this style of project.
- Team members shall not interfere with the standard operations of Dwolla.
- Team members must remain within the defined scope and adhere to the agreed-upon rules of engagement.
- Team members shall document all of their findings and procedures for use in final deliverables.
- Team members must adhere to the signed non-disclosure agreement for information gathered during the course of the test.
- Team members are to disclose any potential vulnerabilities responsibly through the executive summary and technical report.
- Certain attack vectors are to be ignored and considered out of scope. These vectors include denial of service, crawling, spidering, accessibility interference/interruption, resource limitations, resource allocation.

Standards

Penetration Testing Execution Standard

We will be following the penetration testing guidelines as laid out by the Penetration Testing Execution Standard (PTES). These guidelines give a general overview of how to organize your pentest and break it down into distinct phases. The phases laid out by the PTES are as follows: Intelligence Gathering, Threat Modeling, Vulnerability Analysis, Exploitation, Post Exploitation, and Reporting.

OWASP Top Ten API Security Risks

OWASP is an organization that collects information on security vulnerabilities. Each year they compile data on security vulnerabilities and release lists about the most common vulnerabilities pertaining to API's and web vulnerabilities as well as application vulnerabilities. We can use these lists to expedite our vulnerability analysis by searching for common vulnerabilities first.

OWASP Top Ten Web App Security Risks

Similar to the list of common API security risks above, this is a comprehensive list of common web application vulnerabilities. This documentation will be referenced frequently in determining our testing methodology, which tools we will employ, and what vulnerabilities to search for.

NIST Technical Guide to Information Security Testing and Assessment

NIST is a government agency that documents technology standards. They also have released documentation regarding how to assess security vulnerabilities properly. This documentation is incredibly in-depth and offers a lot of information on security research. There is also an entire chapter on penetration testing, which offers information on penetration testing phases and things to look for.

Timeline

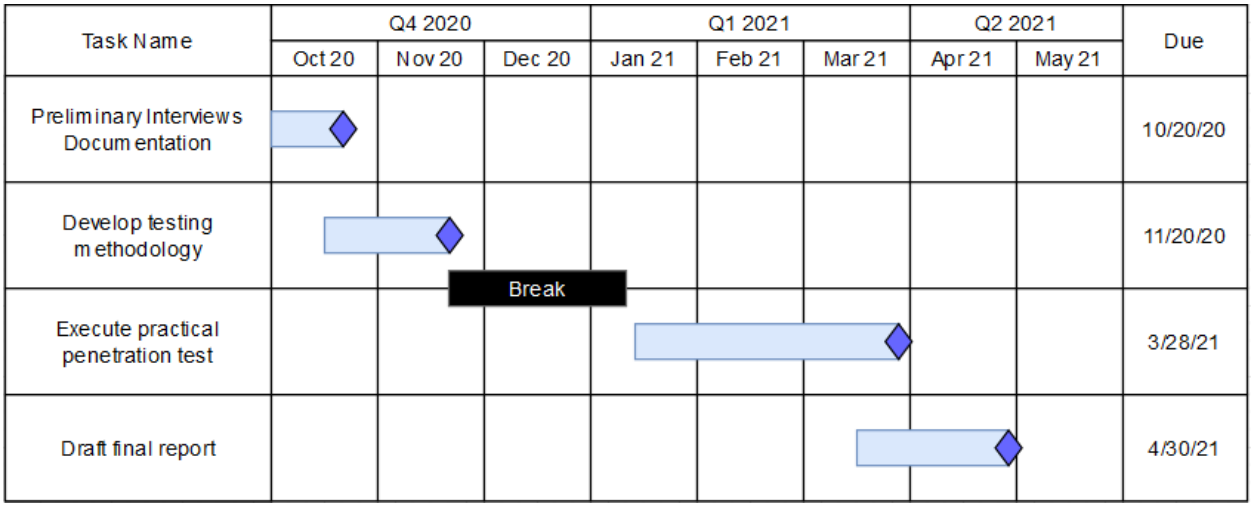


Figure 1.1: A timeline of the project process and plan starting from first semester through the completion of the project.

Preliminary Interviews (10/20/20)

Initial interviews and questions regarding the scope, rules of engagement, timeline, and expected deliverables shall be completed by this date. First drafts of the design document will be completed with details concerning the limitations and expectations of the project. Furthermore, initial research into the OWASP common vulnerabilities and early probing of the given scope will be conducted.

Develop Testing Methodology (11/20/20)

As described above in the task decomposition, the team is expected to have developed a common list of vulnerabilities, attacks, and methods on how the problem area will be addressed. A clear understanding of the tools we will be using, how they will be used, and the steps we will take to complete the pentest is expected by this deadline.

Execute Practical Penetration Test (03/28/21)

At the latest, by this date, the practical penetration test will have wrapped, with details outlining discovered issues, known exploits, and reproduction steps recorded. In addition to these items, a full breakdown of the workload assigned and performed by each team member will be recorded. Lastly, a full severity ranking for each exploit discovered must be completed by this date.

Draft Final Reports (04/30/21)

Finally, with the practical test completed, all information and data recorded during the test will be organized and compiled into a final report for the client. This deliverable will be expected by the end of the semester; however, earlier is preferable to leave time to discuss the found issues, remedies, and possibilities of re-testing to affirm patches. Therefore, a meeting to discuss the final report is expected to happen by the end of April.

Testing Methodology

Limitations

Performing a practical pentest comes with legal and functional limitations as defined by the client. Specifically, all actions we perform must stay confined to the given scope while also satisfying the Rules of Engagement. Our testing methodology will take careful consideration to operate within the boundaries of these preset regulations. Furthermore, all team members are under a Non-Disclosure Agreement contract to maintain the secrecy of certain elements of the penetration test.

Scope

The previously discussed operational area will act as the primary testing environment as we move into the practical execution phase of the project. Team members are to remain within this environment, and the following testing methodology is structured in a manner to meet this requirement.

Rules of Engagement

The previously discussed rules of engagement give a well-defined and structured list of requirements and expectations our team is directed to abide by. These help make the testing process efficient and smooth with limited confusion or disruption to our client. The following testing methodology encompasses and abides by these rules.

Vulnerabilities

- Read through Dwolla provided documentation to determine potential vulnerabilities and document intended and unintended use cases.
- Test web application and API for their OWASP top ten vulnerabilities.
- Search for unintended publicly facing data exposure, misconfigured security controls, poor authentication, or broken access.
- Test API for intended use cases to assess for correct functionality.

- Test API through a variety of unintended use cases to check for incorrect handling of erroneous input.
- Identify possible exploits for found vulnerabilities.
- Evaluate severity for discovered vulnerabilities and exploits.
- Based on risk assessment, develop a set of remediations to properly address the vulnerabilities.

Tools

There are two types of tools that scan or discover vulnerabilities - Web Applications and API. Web application authentication is a solved problem but not for APIs as there are protocols, and it's also common to layer on security requirements. Existing web application security scanners have no concept of any of these standards, and if you manage to get a scanner to authenticate to your API, there's not much luck coercing it. Lastly, APIs aren't discoverable like web applications.

No.	Tool Name	Description	Web App	API	Vulnerabilities
1.	Google Dorking	It is a search technique that enables hackers to gain access to information that corporations and individuals did not intend to make publicly available.	Yes	No	• Sensitive data exposure • Errors of web application • Directory enumeration
2.	OWASP-ZAP	It is a web application security scanner and is a useful way to perform an initial assessment of an application. Offers a large suite of tools useful to finding and assessing web vulnerabilities.	Yes	No	• SQL Injection • Sensitive data exposure • Broken Access Control • Broken Authentication
3.	Postman	Postman is a popular tool for managing and testing API functionality.	No	Yes	• Authentication • Authorization • Broken access • Data exposure • Misconfigured controls
4.	Dwolla SDK	The provided development suite for creating tools which interact	No	Yes	• Authentication • Authorization • Broken access

		with the API. We utilize the Python SDK.			• Data exposure • Misconfigured controls • Compiler issues • Broken code-level functions
5.	Firefox	Popular web browser with many developer and security functions. Includes a packet breakpoint for inspecting and manipulating API/HTTP requests.	Yes	Yes	• Everything
6.	BurpSuite	It is one of the most popular penetration testing, with the community addition offering http capture and packet manipulation.	Yes	No	• Payload injection • Sensitive data exposure • Misconfigured options • DOS Potential
7.	Intruder.io	This tool checks for common misconfigurations for web headers, as well as scans for any outdated services or depreciated options.	Yes	No	• Misconfigured options • Depreciated packages • Cross-Site Scripting • Information leakage

Figure 1.2: A detailed table showing the various tools utilized throughout the testing phase of the project and their application.

Design Evolutions

Project Plan and Methodology

Throughout the duration of our practical penetration test, we have had close communications with our client to provide updates on progress, findings, and exploitability. During these meetings, we also discussed additional tools and vulnerabilities we could research into if we were hitting any dead ends. With this, some of the initial tools we planned on using ended up not being useful in the penetration test (such as Burp Suite and Google Dorking), and newly researched tools proved useful during the test (such as intruder.io and Firefox dev tools). We also found that only one or two members of the team could utilize our single Kali

VM box, so we branched out the testing environment to include personal machines running Kali VMs. Overall, the testing methodology and attack narrative did not change significantly. The Scope, Rules of Engagements, and Requirements for our project remained static throughout the entire duration, and thus overall, our approach mirrored this.

Web Application Testing

When testing both the dashboard and account web application pages, we follow through with our initial research, and tested both web applications first for common vulnerabilities and weaknesses. We used tools such as BurpSuite and OWASP Zap to proxy our web traffic, in order to manipulate both the order and the contents of information being sent to the web servers. We also used automated tools such as intruder.io to get some of the well known misconfigurations tested quickly and efficiently.

API Testing

For the Sandbox API environment, our initial approach vectors remained mostly constant. The team was required to do the expected amount of research and investigation of the API functionality and provided documentation as initially anticipated. Initially, the testing approach for the API included more use and functionality within Burp Suite; however, the free edition of Burp Suite came with some unanticipated challenges. Therefore, the team pivoted away from this and moved more towards using Zap, Postman, and Firefox developer tools to perform testing.

IMPLEMENTATION

Client Information

As the first step within our project plan, we orchestrated numerous interview sessions between the full group and our client, Benjamin Blakely. From these meetings, we fleshed out a fully detailed Scope, well-defined Rules of Engagement, and discussed the operational environment. During this time, we also spoke about tools and methods that would be useful within our operational environment. After initial interviews, we prioritized making sure that each member had a clear understanding of the testing environment we were working in. This included creating user accounts within the sandbox environment and gaining an initial understanding of the services we will be exploiting.

Staging

After concluding the initial reconnaissance and research phase, we moved on to implementing our testing machines. The first step in this process was acquiring our team's Kali Linux box from the Iowa State ETG. After some additional software installation to make using the machine more user-friendly, we went ahead and installed all researched tools that were not already pre-installed on the machine. This included the Dwolla SDK, one of the main tools used in order to test the Dwolla API.

TESTING

Risk Ratings

The following ranking system was developed for our penetration test in order to rank the vulnerabilities found. This system follows guidelines outlined in the Penetration Testing Execution Standard, as well as the CIA triangle, both popular information security models. We adapted the rankings from these standards in order to best fit our client's concerns, which were outlined in the interview phase of our project. This system is also used to give Dwolla staff a face-value rating on the potential dangers that can be caused by the vulnerabilities outlined in our final reporting documents. This will help Dwolla better understand the risks found and assisting in prioritizing in which order these vulnerabilities should be addressed and patched.

Ranking	Description
Critical	*Critical security vulnerabilities represent a severe threat to one or multiple of the CIA triad.* **Confidentiality**: Total loss of confidentiality for one or many accounts, full exposure of sensitive data. Easy access for grand theft and larceny of data. **Integrity**: Massive breach in systems' integrity. Full access to arbitrary/remote code execution, overwrite ability, or full, unauthenticated access to manipulation of stored data. **Accessibility**: Full outages and catastrophic downtime. This means potentially many or all clients would be unable to utilize the functionalities of Dwolla's infrastructure for an extended period.
High	*High security vulnerabilities represent a significant risk to one or multiple of the CIA triad.* **Confidentiality:** Exposure of sensitive data for one or few accounts. Presents possible access points for theft and larceny of data. Possible leaks to allow DOX or targeted attacks. **Integrity:** Possible targeted attacks against integrity on one or part of one system. Potential database manipulation, injection, or arbitrary/remote code execution on a small scale. **Accessibility:** Significant impact to the accessibility of Dwolla services. Remediable full downtime, or specific downtime of individual systems.
Medium	*Medium security vulnerabilities pose a moderate threat to one or multiple of the CIA triad.* **Confidentiality:** Indirect exposure of sensitive data through

	targeted attacks. Represents a slight possibility of data theft and potential for targeted attacks through cross-referencing. **Integrity:** Non-invasive threats to system integrity. May allow some influence over elements or aspects of a system, but cannot impact a system holistically. **Accessibility:** Moderate impact to uptime of minor systems or system elements.
Low	*Low security vulnerabilities contribute little threat to any elements of the CIA triad.* **Confidentiality:** Over exposure of certain elements of systems or users without leaking any sensitive information. **Integrity:** Represents a potential future threat to integrity if left unchecked or in conjunction with another risk. Does not pose an integrity threat alone. **Accessibility:** Potential for impact to system accessibility through specialized tools or resources or in conjunction with another weakness. Does not impact accessibility easily alone.
Informational	Informational security vulnerabilities do not threaten any aspect of the CIA triad. These represent non-trivial areas of improvement or future-proofing for Dwolla's security practices.

Figure 2.1: A table depicting the levels of risk and their justifications assigned to each unique vulnerability.

Testing Process

The following images highlight the network infrastructure map of Dwolla and its services. Furthermore, this highlights how our team connected to this network infrastructure from our personal machines.

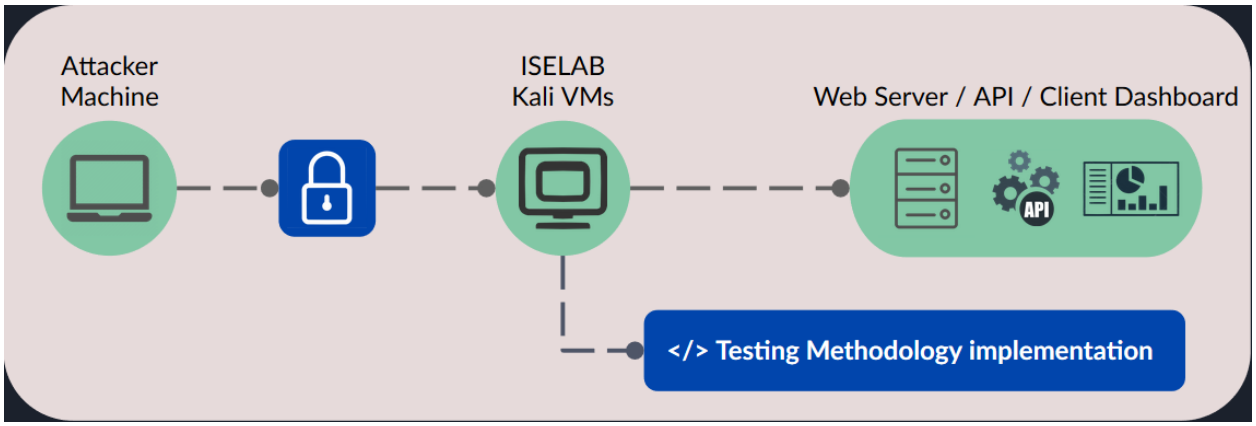


Figure 2.2: Flowchart of the process followed by our team members to access the Dwolla services from our personal machines.

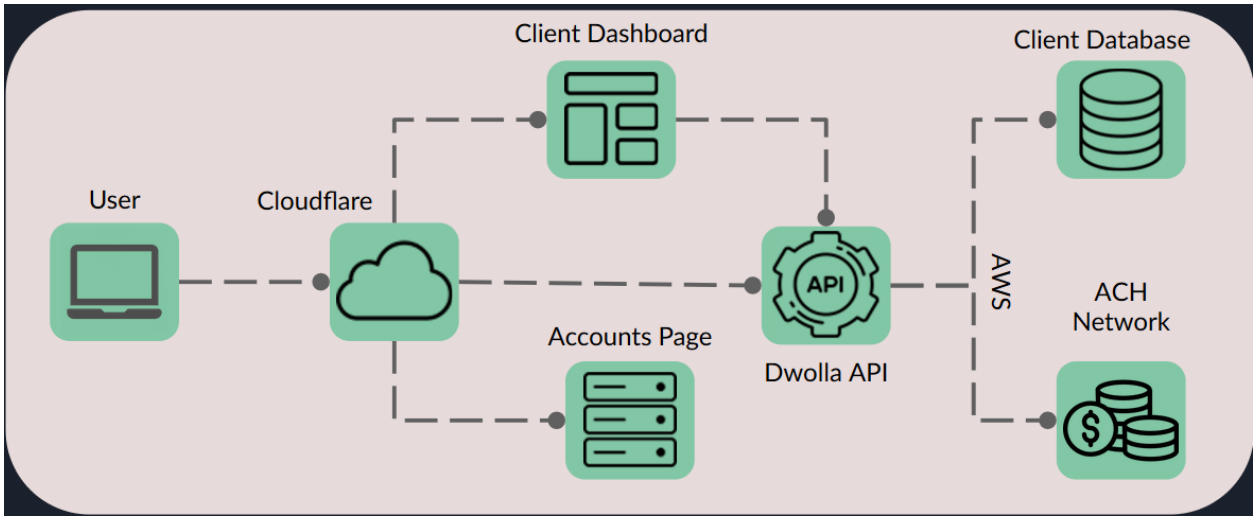


Figure 2.3: Flowchart showing a breakdown of the Dwolla infrastructure from remote user connection to backend services.

Testing Results

As part of our final reports, we were tasked with reporting all vulnerabilities found during our testing. As part of these reports, The following graphics were created for the benefit of Dwolla's business staff to give them an at a glance view of the risk facing Dwolla.

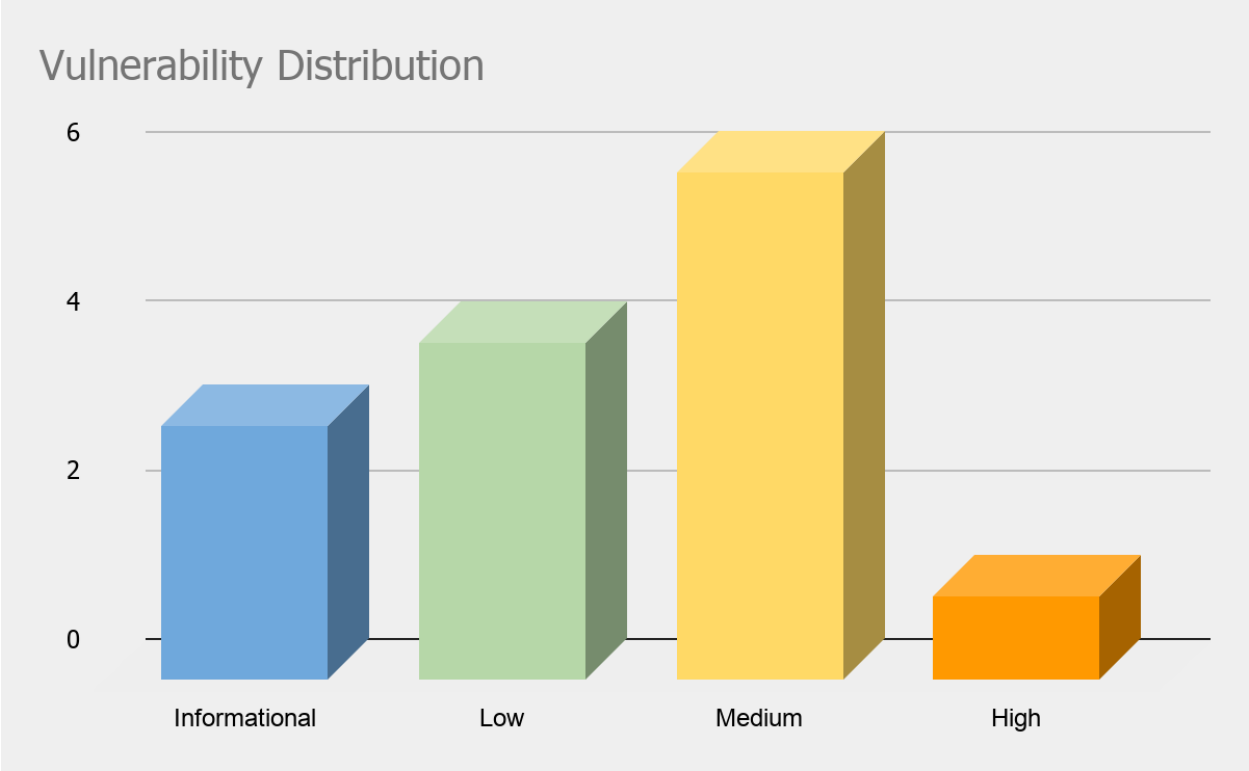


Figure 2.3: Distribution of vulnerabilities based on their rankings found during the penetration test.

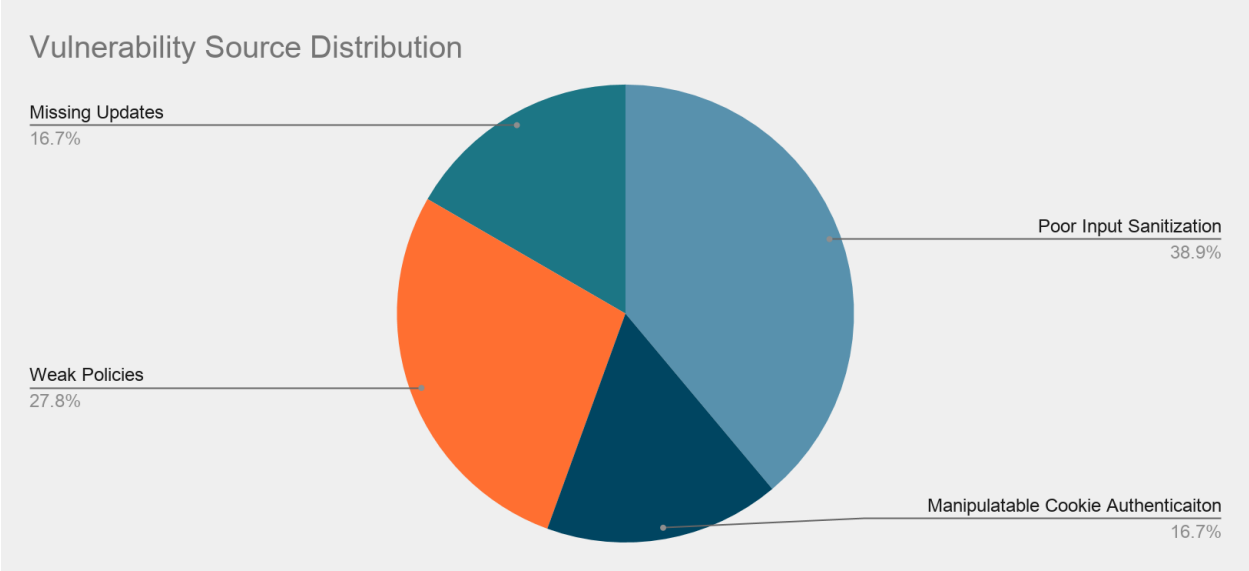


Figure 2.4: Distribution of the root cause of the vulnerabilities found during the penetration test.

Both of these diagrams help represent how many vulnerabilities were found, the distribution of them in respect to their ranking, and a representation of what underlying issues lead to the exploits. For Dwolla's technical staff, we provided a more in-depth and detailed analysis of each vulnerability, which is included in our operations manual in Appendix I.

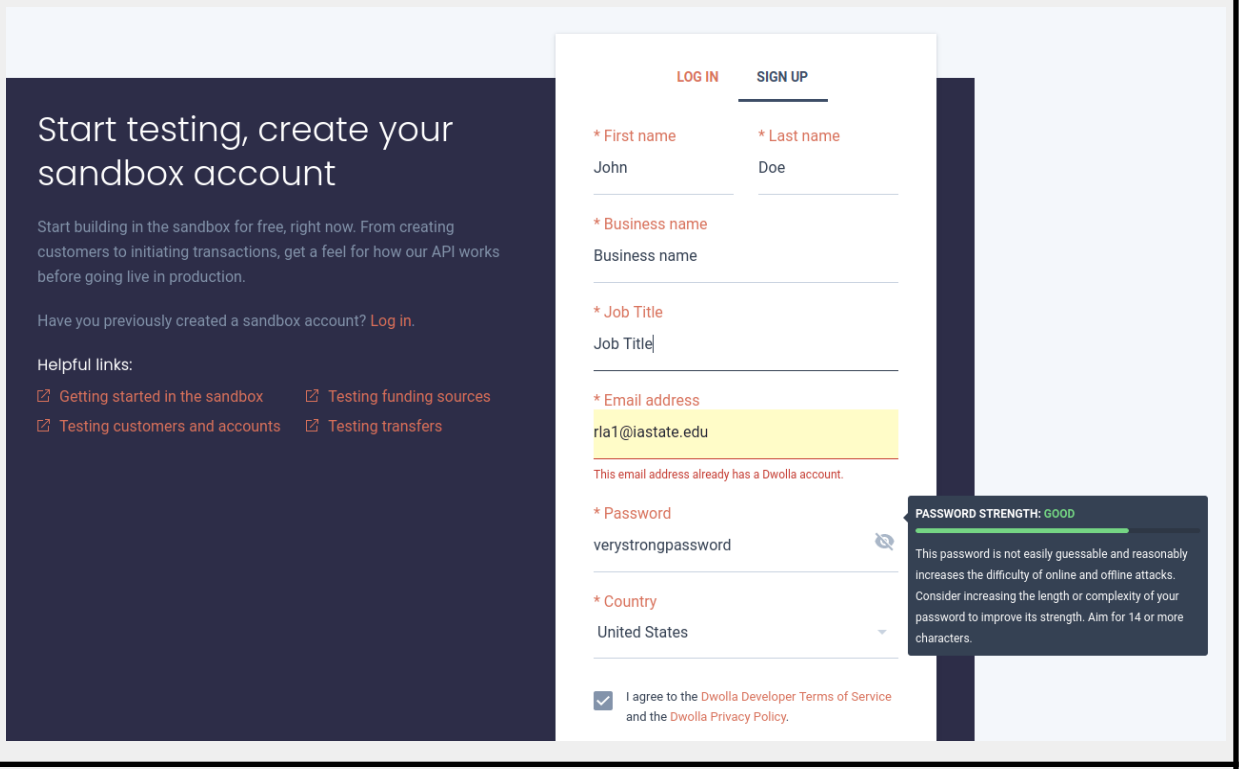
APPENDICES

Appendix I -- Vulnerability Table

For each vulnerability found during the penetration test of the Dwolla sandbox environment, we have created a table that will detail the vulnerabilities description, its assigned risk rating, steps taken to confirm the vulnerability, and advice on how to remedy the vulnerability. These tables act more or less as our operations manual for our client so that they are able to recreate the vulnerability in their environment and have insight on how to patch the vulnerability. Some of the tables have been redacted at the request of our client.

Note: Some of these vulnerabilities are redacted due to their sensitive nature and by request from Dwolla.

User Email Enumeration	Medium
Domains vulnerable: Sandbox accounts page	
Description: The email addresses of Dwolla's clients can be found out through the account creation process.	
Confirmation: The screenshot listed below shows that when attempting to create an account for an existing user, the error "This email address already has a Dwolla account."	



Remediation Advice: In order to resolve this issue, we recommend not erroring out when someone signs up with an existing email address, but instead responding with "If the account creation was successful, you will receive an email regarding account verification and activation". This message would apply to all account creation attempts.

Weak Password Policy	Medium
Domains vulnerable: Sandbox accounts page	

Description: While the password policy still rejects some of the most common passwords, the lack of complexity requirements allows for guessable passwords.

Confirmation: Passwords such as "dwollapassword" and "verystrongpassword" are passable and rated as "good", and listed as not easily guessable.

Remediation Advice: We recommend adding in complexity requirements for special characters, capital letters, and numbers on top of the current system you are using, in order to ensure customer safety.

[redacted]	[redacted]
Domains vulnerable: [redacted]	
Description: [redacted]	
Confirmation: [redacted]	
Remediation Advice: [redacted]	

SID cookie re-use	Low
Domains vulnerable: Sandbox dashboard	
Description: Each time the user interacts with pages within the dashboard, a new Session ID (SID) cookie is generated. Old SID cookies which are no longer used by the browser are still valid to make new requests.	
Confirmation: In order to validate this vulnerability, two SID cookies are needed. An older one which is still within its TTL, and the SID cookie currently used by the browser. In this example, here are the cookies we used: SID 1:	

v3:session#746b429e-1cf6-42c3-b836-82346f70ef85#user#9933926a-095f-4124-bd5e-26afe9b1b613|1617059095280:KIHlnc9DpHx4e0gLUYcJFPP4dxnd-PT4VFcXcvoljLU=

SID 2:
sid=v3:session#746b429e-1cf6-42c3-b836-82346f70ef85#user#9933926a-095f-4124-bd5e-26afe9b1b613|1617059133851:z15NAJwbzbm6AOsTlOJ-kJ6ho1Y92X4RSSGzY8Y9Nok=

Updated request replacing SID 2 with SID 1, and making a request to the server results in a 200 return.

Remediation Advice: Each time the browser will generate a new SID cookie for the user, make sure that all SID cookies not in-use are de-validated, and no longer allowed to make new requests. This will help prevent advanced local cookie attacks.

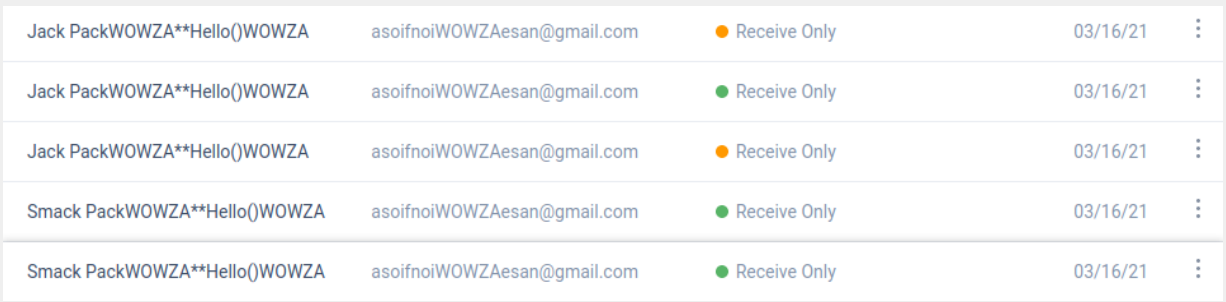
Dashboard Duplicate Accounts	Low
Domains vulnerable: Sandbox dashboard	

Description: With input validation for emails not being processed on the API server, you are able to create customers that appear to have the same email address, as the sandbox dashboard removes all leading spaces.

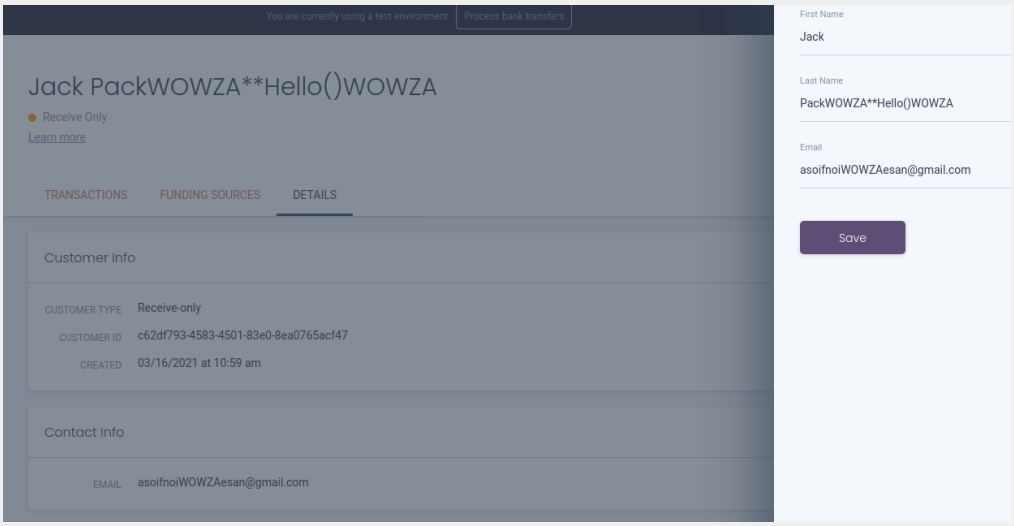
Confirmation:
Here is the text for the customers call, showing that accounts are stored with invalid email addresses, with varying leading spaces

{"list":[
displayName":"Jack PackWOWZA**Hello()WOWZA","email":" asoifnoiWOWZAesan@gmail.com",
displayName":"Jack PackWOWZA**Hello()WOWZA","email":" asoifnoiWOWZAesan@gmail.com",
displayName":"Jack PackWOWZA**Hello()WOWZA","email":" asoifnoiWOWZAesan@gmail.com","
displayName":"Smack PackWOWZA**Hello()WOWZA","email":" asoifnoiWOWZAesan@gmail.com","
displayName":"Smack PackWOWZA**Hello()WOWZA","email":"asoifnoiWOWZAesan@gmail.com"
]}

And listed below is a screenshot of how these email addresses appear in the dashboard:



These email addresses also appear with no spaces on the clients specific page, as well as their reset password tab.



Remediation Advice: We recommend using some checks on the server end to make sure that only valid emails are being passed to the API, as well as making sure the client side displays the true email address you get from the API, as to not allow pseudo duplicate customers.

Multiple Header Suggestions	Info
Domains vulnerable: Sandbox accounts page and dashboard	

Description: These are a combination of multiple upgrades to your http headers. While their absence does not cause any direct security threats, it is a good idea to upgrade, and use the latest technology for security.

Remediation Advice:
The X-XSS-Protection header is no longer supported by modern browsers. As your headers already has its replacement content-security-policy header in place, it would be best to either remove it, or set the value to 0

For the content-security-policy header, the websites have the frame-src option, which is now depreciated. A new replacement option child-src is its updated and more robust option.

The Referrer-Policy header is a newer header, and supported by most modern browsers, and helps govern information that is sent to websites when users click on hyperlinks. It would be a good idea to enable this header to help control sensitive

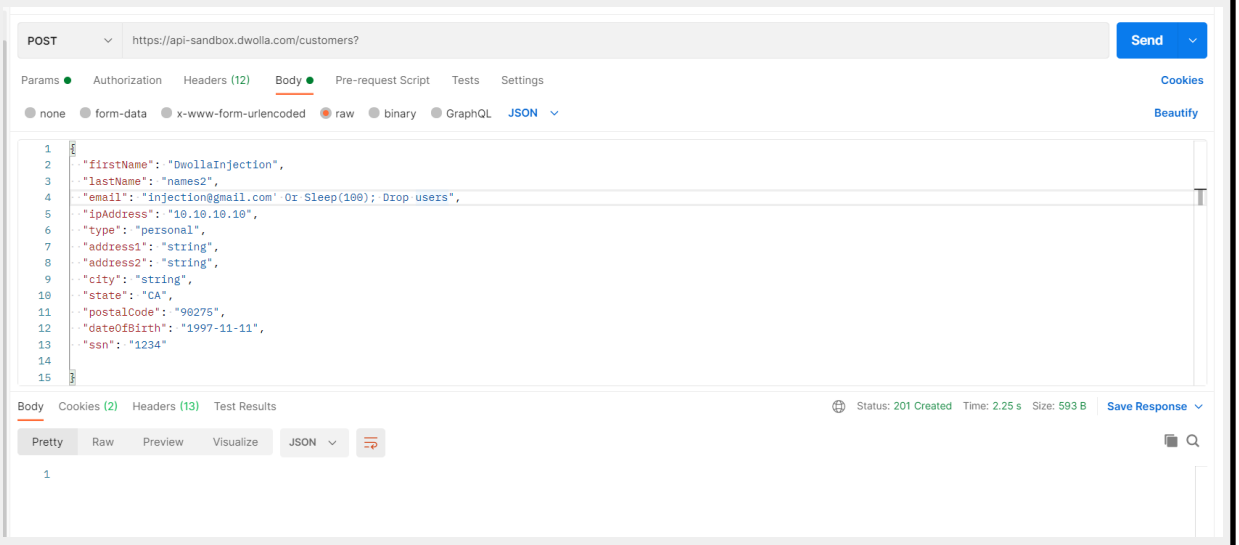
information TLS 1.2 is the current encryption method being used. While this protocol is secure, it has been known to be vulnerable to downgrade attacks in the past. Best security practice recommends upgrading to TLS 1.3, as it is more robust and supported by most major browsers.

Email overflow	Low
Domains vulnerable: Dwolla API	
Description: Because there is no sanitization of the email address server side, you are able to send email addresses with a theoretical infinite amount of characters, and the server will process it, and throw 500 errors	
Confirmation: The request tested for overflowing the email category on the server consisted of a string of around ~1500 characters, the resulting 500 error was sent back to the user: HTTP/1.1 500 Internal Server Error Date: Mon, 29 Mar 2021 18:59:33 GMT Content-Type: application/json; charset=utf-8 Connection: close Content-Length: 107 {"code":"ServerError","message":"A server error occurred. Error ID: e96afbfc-11fc-4e62-9014-207c5d03850e."}	
Remediation Advice: In order to solve this issue, ensure that the API is checking the length of the email string, making sure that it is a size that it can handle. Because all of the information is containerized, you have protection against further overflow attacks.	

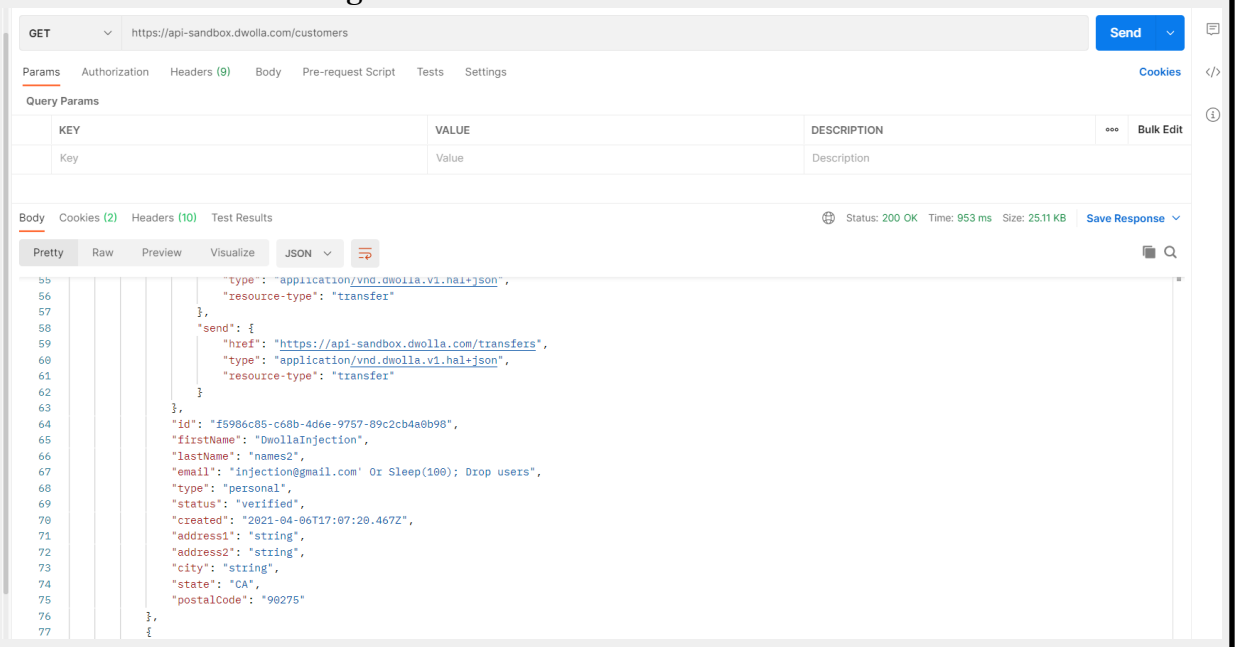
Account First Name Overflow	Medium
Domains vulnerable: Sandbox accounts page	
Description: When creating an account for the sandbox environment, there is no limit on the number of characters in the first name. Using an excessive amount of characters will still create the account, but logging into the account will only return a 500 error.	

Confirmation: For this test, I made an account with the email aSmileyface23232323@gmail.com, and my first name was around 30,000 characters long. Here is the response after logging into the dashboard: HTTP/1.1 500 Internal Server Error Date: Mon, 29 Mar 2021 20:19:13 GMT Content-Type: text/html; charset=utf-8 Connection: close set-cookie: sid=cookie Content-Length: 6466
Remediation Advice: Only allow first names which the servers are able to handle, and limit the number of characters to a reasonable number so that there are no issues in the future.

Input Sanitization	Medium
Domains vulnerable: API	
Description: The customer email field does not sanitize user input through the API. This allows a user to POST a malicious injection in the email field when creating a new user. The backend does not actually read the data POSTed so SQL injection against the backend does not exist. However, malicious emails can then be received through a GET to clients who may not be properly sanitizing input resulting in SQL injections against the client. Although input is not validated on the webapp dashboard, Cloudflare WAF blocks all injection attempts.	
Confirmation: Using Postman we can create a new customer through the /customer endpoint. This returns an http 201 created meaning our customer was created with an email = "injection@gmail.com' Or Sleep(100); Drop users".	



Although this injection is not read and run on the backend, the email still exists and a client can retrieve it using a GET.



Remediation Advice: Sanitize the input coming in from API POSTS. This includes removing whitespace, escaping common SQL characters such as the pound symbol (#) and hyphens (-), which are both used to comment out following logic. Additionally escape full SQL commands such as SELECT, UNION, DROP, SLEEP, as well as common bypasses such as SelEct, %00select, and sel/**/ect.

[redacted]

Domains vulnerable: [redacted]
Description: [redacted]
Confirmation: [redacted]

Remediation Advice:

Encryption Standards	Info
Domains vulnerable: API	
Description: API traffic is communicated with "TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256". TLSv1.3 provides greater support for better cryptography including upgraded ciphersuites such as Curve25519.	
Confirmation: Viewing any Dwolla page with a browser's inspection tool shows the encryption suite utilized. The above suite is what Dwolla employs on their pages.	

Remediation Advice: No part of this ciphersuite is deprecated yet however there are better options that can be implemented such as Curve25519 or SHA3.

Bad Input Queries	Low
Domains vulnerable: API	
Description: It seems that the server accepts requests even with bad query information. Such as “api-sandbox.dwolla.com/?=<script>alert(xss);</script>”. The query is converted to HTML on the backend hence the reason it does not parse into proper javascript, however it is still accepted.	
Confirmation: Copying the exact web URL above will take you to the api-sandbox page with a credential error on the Dwolla network. The credential error signifies the request was accepted, but without proper authentication.	
Remediation Advice: Disallow bad query inputs such as this. Though they are not parsed, they still should not be allowed, and can open up customers/clients to potential attacks.	

Token Expiry	Info
Domains vulnerable: API	
Description: When making requests, the OAuth token handshake occurs every time and a new token is granted. Even though new tokens are requested every time, old tokens are still valid until their expiry.	
Confirmation: Inspecting the request history with a tool like Zap, Postman, or the dev tools on a browser shows that with each request, a new token is requested and generated.	
Remediation Advice: Either set old tokens to expire once a new one is generated for the same user/account, or do not request new tokens until the old one has expired.	

[redacted]	[redacted]

Domains vulnerable: [redacted]	[redacted]
Description: [redacted]	
Confirmation: [redacted]	
Remediation Advice: [redacted]	

Appendix II -- Project Alternatives

Throughout the design and testing phase of our project, the team regularly discussed approach alternatives with our client and advisor Ben Blakely. The bi-weekly meetings acted as an open discussion to brainstorm varying strategies and techniques in testing and design implementations. Coming from the professional arena of cyber security, Ben was able to contribute his experience and expertise in adapting our approach to meet the goals of the project. Early on, our team chose to opt for a free and open-source approach towards testing and tooling. An alternative avenue would have been to allot a budget to our project and permit the purchase and use of certain premium or business-class tools. Higher grade tools such as these would have changed our approach and strategy in implementing our methodology during testing. Furthermore, the automation provided by these tools may have permitted our team to expand the scope and cover more of the environment with a finger comb. Ultimately, we opted against this alternative methodology in favor of a cost-free project. In addition, our original plan focused on performing taught penetration testing strategies against on-premises or virtual hardware/servers. After preliminary discovery on the infrastructure and foundation of Dwolla's network, we learned their servers were virtualized and abstracted behind AWS and Cloudflare. This forced our team to adapt our strategy and pivot to an alternative approach. This revision dictated we apply focus more heavily to the front-end features offered by Dwolla rather than attempting to breach the backend systems. For our team, we lacked experience and expertise in this way, and therefore such a pivot required a bit more research and mastering before testing could get underway. Overall, this pivot worked for the best, and we were successful in following this alternative approach.

Appendix III -- Other Considerations

As a team of future security-minded professionals, what we learned throughout this senior design project will be invaluable to us in the future. As a team, we worked together to perform our first-time penetration test against both a web application and API. This project also gave our team the chance to perform meaningful work to try and better the Dwolla service, both for the benefit of the company and clients. Being able to work with our client, Ben, throughout the process helped give the team confidence that the tools we were selecting and the methodologies we developed were comparable to professional penetration testing standards. Within this team environment, we were also able to learn from each other, as discovering new vulnerabilities throughout the testing process helped teach each member new approaches for evaluating systems in the future. Ultimately, this project has instilled us with valuable knowledge and experience in the security testing world and make us better professionals as we enter the workplace.