Pentest a Payment Provider

Design Document

sdmay21-06

Dwolla - Ben Blakely

The team:
Max Solaro - Chief Pentester
Matt Maiman - Testing Engineer
Ryan Anderson - Lead Reporter
Priyanka Kadaganchi - Facilitator & Scribe
Nathan Key - Editor

sdmay21-06@iastate.edu
https://sdmay21-06.sd.ece.iastate.edu/

Last revision - 11/15/2020

Executive Summary

Development Standards & Practices Used

- National Institute for Standards in Technology 800-115
- Payment Card Industry Data Security Standard
- Penetration Testing Execution Standard
- Certified Ethical Hacker Training
- Rules of Engagement

Summary of Requirements

- Conduct preliminary meetings with Dwolla client to discuss deliverables, milestones, goals, area of operation, rules of engagement, testing methodology, and anticipated vulnerabilities
- Develop comprehensive scope and rules of engagement to follow when conducting practical test
- Perform the practical penetration test following proper development of previous requirements
- Catalogue and report findings to Dwolla client following completion of test
- Perform any necessary retesting after appropriate patch work has been completed

Applicable Courses from Iowa State University Curriculum

- CprE/CybE 230 / 231 / 234 / 331 / 430 / 491 / 532
- ComS 227 / 228 / 252 / 309
- SE 421

New Skills or Knowledge Learned

- Methods to perform a comprehensive penetration test against an API-based application
- Discovery and use of many new security tools, particularly for API testing.
- Expanded understanding of Kali, Linux, and their applications in the security realm.

Tables, Graphs, Images

Table 2.1: Potential risks that may occur during the project, and their mitigation plans.

Table 2.2: Projected timeline for project milestones.

Table 2.3: Team role breakdown including estimated man hours per week.

Table 2.4: Estimated time allotment per every major milestone.

Table 3.1: Researched tools that will be used during penetration tests.

Figure 3.1: Visual aid demonstrating the individual components of the pentesting process.

Figure 3.2: Detailed task decomposition for the three major operational components of the penetration test minus initial planning and methodology development.

Figure 6.1: Pipeline view of the penetration test workflow.

Table of Contents

1 Introduction

1.1 Acknowledgement

Benjamin Blakely -- Professional advisor, client. Contributed significant technical advice, assistance, tools, and references.

1.2 Problem and Solution Statement

Problem Statement

The payment provider Dwolla is contracting a team of cybersecurity professionals to perform a thorough penetration test against the Sandbox API provided by Dwolla for their clients. The cybersecurity actors will perform as Red Team to assess the API, Sandbox dashboard, and Sandbox accounts login for known security vulnerabilities. Because the Sandbox API is public, found vulnerabilities must be validated and risk assessed. It is critical that Dwolla receive every security perspective possible to flush out any security issues and risk present on a public-facing service.

Solution Statement

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 on the problem area assigned by Dwolla and prevent unauthorized security events. During the test, Red Team is expected to discover, validate, and appraise severity of common security vulnerabilities within the defined scope. The goal, therefore, is to identify these potential vulnerabilities within the API 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 and all discovered vulnerabilities with their associated risk audits will be compiled into a final report for the client. Potential remediation approaches will be discussed with Dwolla following completion of the penetration test. Re-testing may be done at a later date if deemed necessary.

1.3 Operational Environment

Scope

We have a very well defined scope for this project that we must work in. We have three URLs which we are to stay within: https://api-sandbox.dwolla.com, https://accounts-sandbox.dwolla.com, and https://dashboard-sandbox.dwolla.com. All of these networks live within a sandbox environment, not a live production. We will be performing these tests from our personal computers and from Kali Linux VM's. While pentesting the web api, we will go through OWASP top ten web application security vulnerabilities. Looking at the dashboard and accounts sandbox, we will focus more on application vulnerabilities.

While testing the sandbox environment, we must ensure that we do not interfere with other users as the sandbox is an open environment. In addition, we mustn't utilize any attacks that would stress the server or perform a denial of service. This includes, crawling, fuzzing, and intentional DOS's. Furthermore, any vulnerabilities found should be responsibly and ethically disclosed to Dwolla.

1.4 Requirements

- A team member is required to stay within the rules of engagement and scope of this project.
- A Dwolla sandbox account: This account will be used by all the team members for penetration testing against the Dwolla network.
- Detailed documentation of any components within the scope should be made available to testers to implement pentesting smoothly.
- Detailed documentation for the client as well for them to know the details and implementation guidelines.
- Security controls that would detect or prevent testing. Consider whether these should be disabled or configured to not interfere during testing.

1.5 Intended Users and Uses

The intended audience for the pen test deliverable is Dwolla and their clients. Dwolla is the primary recipient and audience of the final report deliverables and would receive these deliverables and utilize them directly. Dwolla's clients benefit from the project indirectly from security fixes implemented as a result of the test.

1.6 Assumptions and Limitations

Assumptions

- 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(5 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 a senior project will be adapted to meet the expectations of a pentest.

Limitations

- 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.

1.7 Expected End Product and Deliverables

Objective

The development and execution of a penetration testing methodology. This pentest will provide Dwolla with valuable new perspectives on the security of their API.

Deliverables

- **Scope and Rules of engagement during the pentest**

  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.

  Delivery Date: 10/20/2020

- **Methodology used during the pentest**

  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 API software.

  Delivery Date: 11/20/2020

- **Executive Summary of the Pentest**

  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.

  Delivery Date: 3/28/2021

- **Technical Report of the Pentest**

  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 API and consider how they will remedy the issues based on vulnerability impact.

  Delivery Date: 4/30/2021

2 Project Plan

2.1 Task Decomposition

Develop testing methodology

- Familiarize ourselves with the software that we will be testing
- Perform reconnaissance and gather information on Dwolla's API and Web App
- Research common vulnerabilities and techniques against API's
- Develop list of attacks we will perform on the software

Execute practical penetration test

- Divide work among team to execute different parts of the pentest based on personal research
- Perform planned attacks on Dwolla's client API
- Record and validate vulnerabilities found during the testing
- Attempt further escalation into the software after initial vulnerability
- Record, but do not exfiltrate, sensitive information found during the test

Draft final report

- Gather vulnerabilities and sensitive information found from each group member
- Create high-level report for clients covering general findings and remediation plans
- Create technical report proving vulnerabilities and providing technical details
- Present both documents, as well as presentation, to the client

2.2 Risks And Risk Management/Mitigation

For calculating the risk factor for our project, it will be based on two criteria. These criteria will be likelihood of the risk occurring, and impact of the risk occurring. Each criteria is based on three factors, each of which is rated on a scale from 1 to 3. Those ratings are then multiplied together and divided by 3. The total risk factor is the multiplication of total likelihood and impact, then divided by 2. Any risk over 5, we have developed a mitigation plan.

Risk Likelihood Factors

- Ability to cause the risk
- Motivation to cause the risk
- Knowledge of systems required

Risk Impact Factors

- Loss of confidentiality
- Loss of integrity
- Loss of availability

Potential Risk	Risk Factor	Mitigation Plan
During the execution of our pentest, a risk exists that we will not be able to find any vulnerabilities significant enough in Dwolla's API .	Likelihood = 5 Impact = 7 Risk Factor = 6	Our team will first search new tools and develop additional strategies for testing Dwolla's API. If that fails, we will test our methodology against vulnerable API's popular for testing common vulnerabilities.

The team may expose a vulnerability that could cause major damage to Dwolla.	Likelihood = 1 Impact = 9 Risk factor = 5	Our team will notify Dwolla immediately and provide help to patch this vulnerability, rather than waiting until the reports are written
Team members' personal IP addresses could become blocked by Dwolla's attack prevention systems.	Likelihood = 2 Impact = 2 Risk factor = 2	N/A
Team members lose computers / are unable to connect to the world wide web.	Likelihood = 2 Impact = 4 Risk factor = 3	N/A

Table 2.1: Potential risks that may occur during the project, and their mitigation plans.

2.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Develop testing methodology (11/20/20)

- Familiarize ourselves with the software that we will be testing. This will be done by reviewing the Dwolla provided SDK. Additionally this can be considered completed after an account is created within the Dwolla Sandbox environment and fund transfers are simulated correctly.
- Perform reconnaissance and gather information on Dwolla's API and Web App. This can be considered completed after we have successfully simulated requests against Dwolla's API using Postman or a similar tool.
- Research common vulnerabilities and techniques against API's. This can be considered completed after reading through OWASP's top ten web and application vulnerabilities .
- Develop a list of attacks we will perform on the software. This can be considered completed after a list of vulnerabilities is composed.

Execute practical penetration test (3/28/21)

While executing the penetration test, we will use the list of potential vulnerabilities previously identified. Progress can be tracked as each vulnerability is evaluated. If no exploits are found for a vulnerability, said vulnerability can be documented as tested for and we will continue moving down the list. In the event an exploitable vulnerability is found, we will document it and potentially attempt to escalate privileges.

Draft final report (4/30/21)

- All reporting will follow the standards laid out by PTES or the Penetration Testing Execution Standards.
- The first part of this report will cover the intelligence gathering phase of the pentest. This includes all information obtained through passive and active intelligence gathering as well as information given to us by the client.
- The next phase is the Vulnerability Assessment. This includes assessing the risk for each vulnerability that is found. Risk assessments are on a predetermined scale of 1-15. In addition to risk assessments, there will be general findings, recommendations and a strategic roadmap documenting any issues found along with solutions.

- Next we will cover Exploitations and Vulnerability Confirmation. The will also include our entire testing methodology, including general vulnerability assessment and post exploitation findings.
- After exploitation and vulnerability confirmation, we will cover post exploitation. This will include any access or information that we were able to obtain as a result of an exploit.
- Next is Risk/Exposure. This section of the report will cover the risks associated with each vulnerability found. This would include the severity of the vulnerabilities and the level of skill required to act on it.
- The last chapter of our final report would be the conclusion. This would be a general overview of our findings and guidance on ways to resolve some of the vulnerabilities found.

2.4 Project Timeline/Schedule

Task Name	Q4 2020			Q1 2021			Q2 2021		Due
	Oct 20	Nov 20	Dec 20	Jan 21	Feb 21	Mar 21	Apr 21	May 21	
Preliminary Interviews Documentation									10/20/20
Develop testing methodology									11/20/20
Execute practical penetration test									3/28/21
Draft final report									4/30/21

Break

Table 2.2: Projected timeline for project milestones.

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 (3/28/21)

At the lastest, 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 report (4/30/21)

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.

2.5 Project Tracking Procedures

- Our group will use Trello which is a list-making application and each card can be appointed to a team member. Which will help us track information regarding vulnerabilities. Google docs will be used to track progress on developing the pentest throughout the course of this and next semester and this will help our team make sure we are not duplicating work.
- For normal communication with the project members we have decided to use discord. Discord is an instant messaging application.
- To communicate with our Client (Ben Blakely) we will do it through email in addition to bi-weekly meetings..

2.6 Personnel Effort Requirements

As this is a 3 credit class it means that about 9 hours should be given by each individual.

Name	Work/ Role	Number of hours per week	Total number of months for this project
Matthew Maiman	Testing Engineer	9- 12 hours	8 months
Ryan Anderson	Lead Reporter	9 - 12 hours	8 months
Max Solaro	Chief Pentester	9 - 12 hours	8 months
Nathan Key	Editor	9 - 12 hours	8 months
Priyanka Kadaganchi	Facilitator/ Scribe	9 - 12 hours	8 months

Table 2.3: Team role breakdown including estimated man hours per week.

Chief Pentester: Primarily responsible for overseeing and conducting the penetration test. Will delegate tasks, oversee their completion, and maintain a master list of vulnerabilities to be tested and vulnerabilities discovered.

Testing Engineer: The testing engineer ensures all expectations of the penetration test are met, assists others with their tests if in need of assistance, and helps the chief pentester whenever needed.

Lead Reporter: The final report at the end of the year will be supervised and delegated by the lead reporter. Also, this individual is largely responsible for ensuring all bi-weekly reports are completed and submitted on time.

Editor: Responsible for reviewing and proofreading all written work prior to submission. Ensures documents are submitted in a professional manner, absent of spelling and grammatical issues.

Facilitator/Scribe: Schedules meeting times for the group and facilitates meetings with our client and faculty advisor to discuss and review our work. Furthermore, records notes on each meeting when necessary.

Tasks time distribution:

Task	Time
Preliminary interviews and research	**2 months**
Develop testing methodology	**2 months**
Execute practical penetration test	**3 months**
Draft final report	**1 month**

Table 2.4: Estimated time allotment per every major milestone.

2.7 Other Resource Requirements

A computer or some device capable of connecting to the testing environment.

Virtual machine or some Linux operating system that can access the necessary tools.

Open source cyber security penetration testing tools.

Access to document and project management software such as Google docs and Trello.

A suitable network connection.

Access to Dwolla testing environment.

Access to documentation including Dwolla API, scope, and rules of engagement.

2.8 Financial Requirements

This project will be utilizing open-source and free software to fully perform the required tasks. No fees or costs will be incurred.

3 Design

3.1 Previous Work And Literature

See 6.2 References for citations.

Penetration Testing Execution Standard

- We will be following the pentesting 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 Analysis

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

Approach

Our project approach departs from other projects in the Senior Design environment in many ways. While the significant majority of projects in Senior Design focus around the delivery of an end-product, either physical or digital, our project focuses on the completion of a service. This creates a distinct division between our project approach, dynamic, and workflow. Rather than having rigid and structured requirements, we have a looser guideline to follow in the Scope and Rules of Engagement. Furthermore, while most projects will have a visual block or unit diagram to accompany their design, whereas because our project does follow a product deliverable, our diagrams consist of network infrastructures, attack method flowcharts, or tables of tools and their applications.

For the project approach itself, we are also needing to largely depart from an operational environment in which the team is comfortable. Previously, our team has been trained in an on-premise or simulated physical environment. Whereas for Dwolla, their environment is cloud-oriented with only an API and web dashboard servicing clients. Therefore, our approach must adapt to this foundation, requiring our team to reassess our abilities, research new tools, and study different breaching techniques.

3.2 Design Thinking

Define phase

- The penetration test is centered around Dwolla's sandbox web application and API in order to find vulnerabilities and protect clients
- Dwolla wants the penetration test to be non-destructive, as to not interfere with their clients' daily operations
- Stay within the bounds of technical standards and ethical hacking, and do not publicly compromise any information about Dwolla or their clients

Ideate phase

- Research and implement relevant software and attack vectors that can be used to find vulnerabilities in Dwolla's systems
- Ensure that all attacks and tools used against Dwolla's web server and API are non-destructive
- Gather more technical information and perform reconnaissance on servers defined in the scope
- Define tools that will be used and common vulnerabilities that are relevant for web application and APIs
- Ensure reporting has both technical and non-technical descriptions and severities of vulnerabilities found

3.3 Testing Methodology

Scope and Rules of Engagement

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. This is why it is paramount to outline these items before the practical testing steps can be defined.

Scope

Our problem area which we will be testing against and confined to is as follows:

The Dwolla Sandbox environment including the given API and associated development tools. This range encompasses all provided sandbox web pages such as the accounts page, control panel, and API documentation as well as any sandbox web pages we find publicly accessible. Furthermore, this covers the sandbox API functionality with the inclusion of the six SDK tools written in NodeJS, Ruby, Python 3, PHP, C#, and Kotlin. Postman is also an option to interface with and test API intended use cases. The sandbox environment should not be exited at any point during testing.

Rules of Engagement

- Remain within the scope of the project at all times. Do not exit the scope into production under any circumstances. Points of access must be reported and documented immediately.
- Assess all possible vulnerabilities, but bare in mind resource allocation. Do not oversaturate the servers and do not attempt any sort of exploit which may compromise the availability of the sandbox service.
- Document, report, and detail reproduction steps for every discovered vulnerability. Do not take steps to correct or patch any vulnerabilities without prior approval from the client.
- Do not exploit anything which may result in lasting damage or otherwise violate other rules of engagement.
- Adhere to the technical and ethical standards of pentesting as defined by NIST.

- Cease any action prior to compromising the integrity of the sandbox environment. Report points of integrity weakness immediately. In this instance, integrity refers to the functionality and operation of the sandbox client including the integrity of the source code and backbone of the environment (i.e. do not inject malicious code or compromise the functionality of the environment).
- If points of confidential information leakage are found, do not view or access such data, rather report the infraction immediately.
- Details of the vulnerabilities found within the Dwolla environment may not be discussed publicly per the NDA specifications. Only after approval with our client may these details be disclosed.

Vulnerabilities

- Read through Dwolla provided documentation to determine potential vulnerabilities and document intended and unintended use cases.
- Test web application for the OWASP top ten and PCI DSS vulnerabilities.
- Search for unintended publicly facing data exposure.
- Test API for intended use cases to determine proper functionality, and to search for unintended leakage of data.
- 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.

Below in the table are listed some popular tools that are used to scan or discover vulnerabilities.

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	• Hidden files on a web server • Back copies • Errors of web application • Login and password
2.	OWASP-ZAP	It is a web application security scanner and is a useful way to perform an initial assessment of an application.	Yes	No	• SQL Injection • Sensitive data exposure • Broken Access Control • Broken Authentication
3.	Netcraft	It is an internet security audit performed by security professionals.	Yes	No	• Cross-site scripting • SQL Injection

					• Server misconfigurations. • Local file inclusion.
4.	JMeter	It is used for load testing purposes and security issues as well. It can work with CSV files which helps in producing unique parameter values for the tests.	No	Yes	• HTTP response splitting • File Inclusion • Memory corruption • SQL Injection
5.	Vooki	It is a RestAPI Vulnerability Scanner and has a deep search algorithm which does advance checks for vulnerabilities.	No	Yes	• SQL Injection • Command Injection • Missing security headers • Improper HTTP Response
6.	BurpSuite	It is one of the most popular penetration testing and vulnerability finder tools, and is often used for checking web application security.	Yes	No	• SQL Injection • File Path manipulation • HTTP Response header injection • Unidentified code injection.

Table 3.1: Researched tools that will be used during penetration tests.

3.4 Technology Considerations

API Testing Tools

Postman is an application that allows a user to send custom http requests to a server. This can be beneficial for us as we test the API's usability and attempt to gain access to sensitive information. Using Dwolla's documentation provided on their website, we can create custom GET/POST/PUT requests and test the api.

The Dwolla SDK is another tool that we can use to test Dwolla's API. The sdk will provide the same functionality as Postman, but it has an additional benefit. When using Postman, we would have to manually enter an authentication token every 60 minutes in order for the API to respond. The DwollaSDK on the other hand would automatically authenticate. Both of these applications will be extremely useful and we will have to test both of them in order to assess which will work better.

Vulnerability Scanners

There are many vulnerability scanners that we can use to scan Dwolla's servers. Nessus is a popular enterprise scanner that has a free license. In addition to Nessus, nmap is a commonly used vulnerability scanner that can be deployed through a metasploit console. Nmap can be used very modularly which can allow the user to scan for as little or as much information as we want. The main issue with nmap is that it can be referred to as a loud scanner, often triggering intrusion detection systems and that could potentially make our ISP's upset.

In addition to these application vulnerability scanners, we can also look into API vulnerability scanners. These vulnerability scanners can check a REST API for common vulnerabilities and weaknesses that we can then further investigate manually. Some good choices look to be NetSparker, Acunetix, and Rapid7.

Virtual Machines

Our pentest will be conducted entirely online through virtual machines. Kali Linux is a distro that is commonly used by red teams and pentesters as it interfaces with metasploit and other tools very well. We will be using Kali boxes for most of our work. The main benefits of using a virtual machine instead of personal machines is that we will be able to test from a constant, static IP. This can help Dwolla to confirm who is testing them. Additionally, testing from an Iowa State IP will remove any suspicions from our personal ISPs. The main con from solely using these VM's is it doesn't allow us any on-prem access to Dwolla's network which would allow for the possibility of many more security threats. However, this is not a part of our scope so VM's will work sufficiently well.

3.5 Design Analysis

The general principles and approach of our design philosophy has remained mostly unchanged through the duration of the project thus far. However, in our initial planning we assumed there would be on-premise network infrastructure for Dwolla, and therefore we would be able to access physical machines or possibly drop into a shell within the Dwolla network. After further research and discussions, we discovered this was not the case and in fact Dwolla operated a wholly cloud-based network with only the ACH batch request system, database, and API functionality. This changed our approach and testing methodology. We had to reassess which vulnerabilities we would be analyzing, and discover new tools to employ against the network.

Ultimately, our general goals, deliverables, and milestones have remained unchanged, but the specifics in how we will approach the practical implementation of this project have evolved. Our development and planning process has also evolved throughout the semester. Initially, the team was strictly following our outlined roles as discussed in section 2.6. However, as we have developed tasks, delegated work, and realized each member's skills, these roles have morphed and changed into what suits each member. From this point forward, we do not anticipate any further major changes to our design plan.

3.6 Development Process

For the duration of the project we will be utilizing the Agile development process to meet assignment deadlines and complete project requirements. Meetings will be held tentatively on a weekly basis to determine areas to complete as well as determining group member assignments for each area. Members will then complete their assigned task(s) before the scheduled due date, or by the beginning of the next meeting, whichever comes first.

3.7 Design Plan

This project involves penetration testing a payment provider. It helps with deep coverage of applications/networks to find high-risk vulnerabilities that are confirmed exploitable.

Below in **Fig. 3.7.1** we see different steps that go for manual penetration testing.First we collect the data from the company then do a thorough vulnerability assessment. Next step is to exploit and try penetrating into it in different ways. This step gives us knowledge about how the data can be exploited. Lastly, we report the preparation.

In **Fig. 3.7.2** We have explained the above diagram with greater detail.

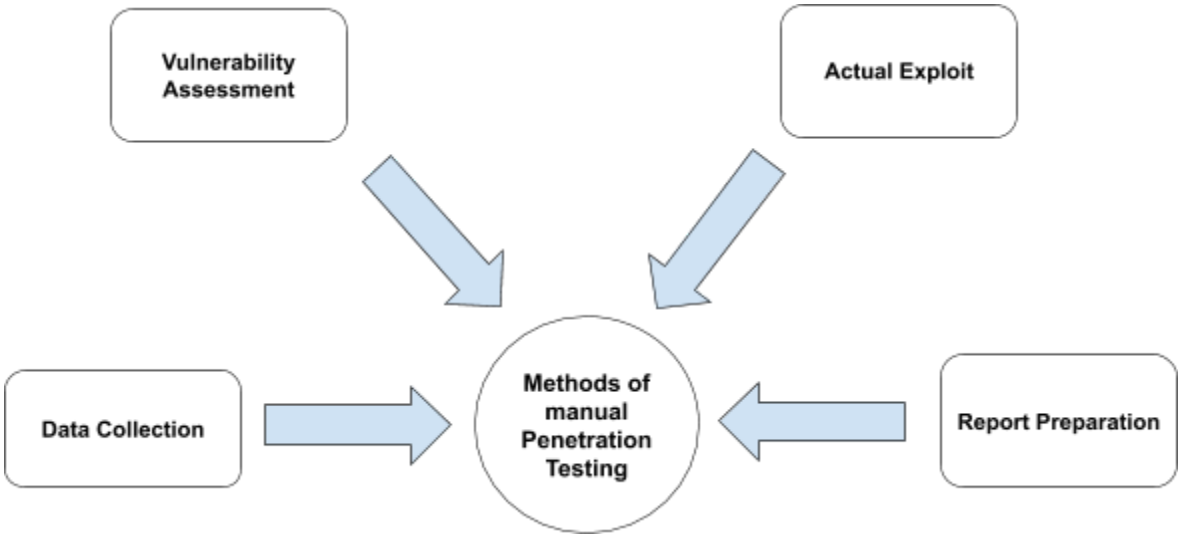

Figure 3.1: Visual aid demonstrating the individual components of the pentesting process.

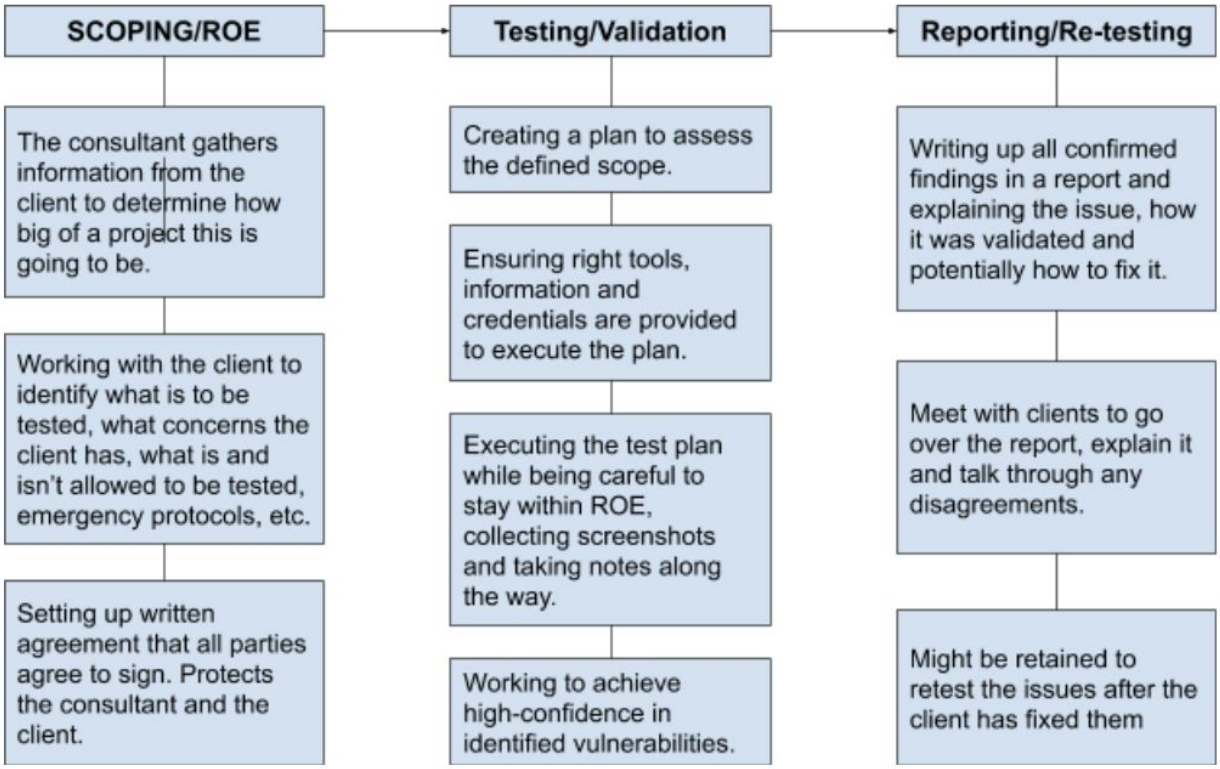

Figure 3.2: Detailed task decomposition for the three major operational components of the penetration test minus initial planning and methodology development.

4 Testing

4.1 Unit Testing

Our defined scope illustrates two primary areas of focus for our testing: the web application which interfaces with the Dwolla API and clients, and the public API which permits clients to create their own tools to interface with Dwolla's services and functionality.

Dissimilar from the standard testing model as with a product or software-oriented project, our testing is the primary purpose and deliverable of this project. Therefore, the testing will not be performed ahead, rather it will be performed as the final implementation. Test results will not be reported in this section, but tests will be properly defined and expectations outlined.

1. **Public-facing Web Application**

   The OWASP top ten web app vulnerabilities defines a comprehensive list of the most common vulnerabilities to date which permeate most web applications today. PCI DSS 3.0 further details best web application practices for payment exchangers such as Dwolla.

   The given testing targets consist of three web addresses which are visible to public clients. These are: https://api-sandbox.dwolla.com ; https://dashboard-sandbox.dwolla.com ; https://accounts-sandbox.dwolla.com. Each of these web pages will be assessed for vulnerabilities in accordance with the OWASP top ten list as well as the PCI DSS best practice recommendations for web applications. We will also employ web crawling tools and scanners to discover unlisted or unindexed pages. We do not anticipate critical security vulnerabilities, however a wide range of tests will be conducted in order to cover every possibility. We do expect to discover minor problems such as outdated software, extensions, or packages, misconfigurations, or even possibly improper visibility to sensitive data.

   To assist with the web crawl, we are authorized to create a testing account within the Sandbox environment. This will also us to poke around the dashboard, any visible web pages, and any unlisted pages. Furthermore, we can test against the access configuration to ensure the accounts cannot elevate themselves or gain unwanted access. This stage will move into the interface testing category as we anticipate possible leaks between the production and sandbox environment.

   OWASP defines the top ten most common web app vulnerabilities which we will be testing against as: https://owasp.org/www-project-top-ten/

   - SQL/database injection to expose or manipulate data tables
   - Broken authentication to leak, hijack, or bypass authentication sensitive data exposure
   - Xml external entities in which xml input with reference to an external entity is parsed and processed which may expose backend information or allow request forgery
   - Broken access control which may permit unauthorized access or operations
   - Security or package misconfiguration
   - Cross-site scripting attack which exploit script languages such as js to inject malicious code
   - Insecure deserialization which may allow root elevation or remote code execution
   - Third-party entities or included components with pre-existing vulnerabilities
   - Insufficient logging and monitoring

   These are the items that will be tested on the web application portions of the scope.

2. **Public API**

   The OWASP top ten API vulnerabilities defines a comprehensive list of the most common vulnerabilities to date which permeate most API tools/interfaces today.

Documentation on the provided Dwolla API can be found at the given web address: https://api-sandbox.dwolla.com/doc and https://developers.dwolla.com/. Dwolla also provides SDKs for various languages which permits tools development in reference to the API. Using either the provided Python SDK or Postman, we will develop bare tools to test the API function and intended use cases within the sandbox Dwolla environment. Utilizing the OWASP top ten vulnerabilities, we will test against some of the common security issues or leaks which permeate APIs today through use of the SDK or Postman tools. As with the web application portion, we do not anticipate major security flaws, however we expect slight excessive data exposure or minimal security misconfigurations. The most essential problem areas we will be investigating involve broken authentication, improper access/elevation, and production exposure.

OWASP defines the top ten most common API vulnerabilities which we will be testing against as: https://owasp.org/www-project-api-security/

- Broken object level authentication in which a user's access level should be affirmed at every function which reaches a data endpoint.
- Broken user authentication which may allow hijacking accounts, bypass auth, or spoofing tokens.
- Excessive data exposure which may over publicize object properties.
- A lack of resources and rate limiting which may permit denial of service attacks through excessive requests.
- Broken function-level authorization in which there lacks a clear distinction between user-level and admin-level functions.
- Mass assignment of properties to data models without whitelisting, simple security misconfigurations.
- Database injections
- Improper asset management which may over expose data endpoints or pre-existing network assets.
- Insufficient logging and monitoring to trace attacks or breaches. These are the items that will be tested on the public facing API

3. **Production Environment**

Aside from our scope within the Sandbox environment of Dwolla, the company operates a primary production environment. We will not be testing any vulnerabilities or web pages within the production environment, however it is important for us to acknowledge it. We will be testing inside the sandbox environment for leaks or loose connections to or from the production environment. These are critical issues and must be reported immediately. This will be discussed more in 4.2 Interface Testing.

4.2 Interface Testing

1. **Sandbox and Production Environment**

Dwolla operates both a sandbox and production environment for their API. For this penetration test, we will not be directly attacking the production environment for the Dwolla API. While we are not testing the production environment, we know that we should not be able to reach any information from the production environment from the sandbox environment. This test will ensure that the two environments do not have any leaks or paths between each other, as that would compromise a clients' information. The implications of this vulnerability existing are severe, as anyone on the internet could have access to Dwolla's clients' sensitive personal and financial information. This test will help verify the security measures put in place by Dwolla that ensures the production environment has no connections to the sandbox environment.

2. **API Interface**

   The Dwolla API acts as an interface between the clients' applications and the Dwolla platform, as well as the interface between the client dashboard and Dwolla's backend systems. The API is responsible for protecting the integrity of the information that is passed between the two systems. Any compromise to this transfer could lead to further damage to both the clients and Dwolla. The tests performed will try and expose and exploit any vulnerabilities in the API that may lead to unwanted information being leaked.

4.3 Acceptance Testing

All investigatory actions, successful penetrations, and vulnerabilities discovered will be reported with full detail on reproduction in the final report for Dwolla. Additionally, each discovered vulnerability will be appraised for an appropriate severity risk and potential security impact. This will demonstrate that each aspect of the given scope, defined unit tests, and methodology has been properly tested and assessed. The report will be well-documented, formatted, and clearly written. Delivery will be expected by April 30th of 2021, following the completion of the practical pentest.

Proceeding delivery of the final report and acknowledgment of pentest completion, we will host a meeting with our client for Dwolla to outline and discuss the report. Any outstanding issues can therefore be reconciled and patched at Dwolla's wish in addition to further re-testing for validation.

4.4 Results

Tentative. - Formal penetration test will not be conducted until Q1 2021. Cannot evaluate results until then.

5 Implementation

5.1 Client Information

Thus far, we have orchestrated numerous interview sessions between the full group and our client, Benjamin Blakely. From these meetings, we have fleshed out a fully detailed Scope, well defined Rules of Engagement, and discussed the operational environment. Due to our discovery period with Ben, we have concluded what tools and methods we will use to approach the problem area. These items have been previously discussed above.

Moving forward, we will take our knowledge of the testing environment in conjunction with the defined limitations and requirements which have been requested of us and apply this to our methodology. This follows into 5.3 Staging where we discuss Virtual Machine (VM) acquisition, how we will prepare and stage the VM, and where our testing methodology will be implemented. Thus, with all of this information documented and recorded, the information discovery phase of the project has been successfully completed and implemented into our design.

5.2 Research

In order to gain a better understanding of the various vulnerabilities that we would be testing for, the team began conducting research into a variety of general areas. Our main focus areas for research were, Common webapp vulnerabilities, API vulnerabilities, Intelligence gathering, networking, and tools. Each team member was assigned one topic to conduct research on and document their various findings. Throughout this process we developed a set of tools we intend to use for the penetration test, as well as narrowed down the vulnerabilities we will be searching for by eliminating network vulnerabilities from our list due to networking being outside of the scope of the test.

5.3 Staging

After acquiring our Kali Linux boxes from Iowa State ETG, there was a bit of initial setup that was required. Firstly, we could only access our vm using ssh and running our entire pentest through a terminal was not going to work well. We really needed a GUI for testing the API and dashboard through a browser so we had to setup xrdp through xfce so that we could rdp into our VM and work with a GUI. Next came a short proof of concept of some of the tools we are going to be utilizing for the pentest. This includes the Dwolla SDK, Metasploit, and Nessus.

The Dwolla SDK was relatively straightforward to set up. Dwolla offers an open download on their website and after installing a few dependencies, we had the python version up and running and ready to simulate API requests. Since we chose Kali Linux as our VM OS, metasploit came pre-initialized along with nmap, so we were up and running quickly. Lastly, Nessus offers free licenses for education so we are currently in the process of obtaining said license. Once that goes through we should have the potential to scan for vulnerabilities utilizing both Nessus and nmap. We will continue to do research on potential API vulnerability and weakness scanners to implement as well.

6 Closing Material

6.1 Conclusion

The project here is to:

1. Accomplish a research and develop an exhaustive testing methodology.
2. Perform a comprehensive penetration test for the Payment Provider Dwolla.
3. We will be documenting and reporting discovered vulnerabilities.

Goals:

1. Secure client systems and services.
2. Protect sensitive data and customer information.
3. Create a trusted environment for money exchanges.
4. Follow PCI-DSS and NIST security standards for payment providers.

Plan of action:

Below(Fig 6.1) is the best plan of action our team developed for penetration test workflow.

Each one of the penetration testers will connect to our Kali Linux VM machine on ISELAB's network through Iowa State's VPN. This will be the main machine used to attack Dwolla's web server, API, and client dashboard.

Using this virtual machine allows the team to operate using a single set of tools, as well as attack from a single IP address, as to not alarm our client. We chose to use Kali Linux as our machines operating system because it is pre-loaded with many tools that we will use during the test, as well as can install the tools we have added.

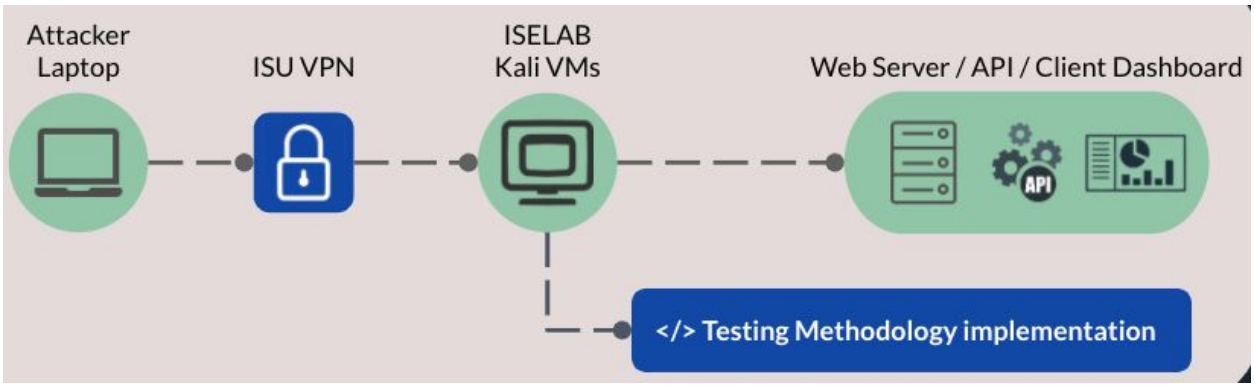

Fig 6.1: Pipeline view of penetration test workflow.

Solution: Identifying potential vulnerabilities will be done using several methods-

1. Targeted vulnerability testing will be done by cross referencing OWASP's top ten web application vulnerabilities and top ten API vulnerabilities.
2. Broad sweep testing will be accomplished utilizing multiple vulnerability scanner tools including nmap and nessus.
3. Dwolla SDK will allow us to test the Dwolla API by forming custom API requests and testing for expected output.

6.2 References

"*Penetration Testing Execution Standard*", Aug. 2014, www.pentest-standard.org/

OWASP Foundation. "*API Security Top 10 2019*." OWASP, 2019, owasp.org/www-project-api-security/

OWASP Foundation. "*Web Application Security Top 10 2019*." OWASP, 2019, owasp.org/www-project-top-ten/

U.S. Department of Commerce. "*Technical Guide to Information Security Testing and Assessment*." National Institute of Standards and Technology, Sept. 2008, nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-115.pdf

PCI SSC. "*PCI Security*." PCI Security Standards Council, 2020, www.pcisecuritystandards.org/pci_security/

Dwolla API references. https://api-sandbox.dwolla.com

Dwolla Sandbox Account Login and Registration. https://accounts-sandbox.dwolla.com

Dwolla Client Dashboard. https://dashboard-sandbox.dwolla.com

6.3 Appendices

N/A