DinoBank Penetration Test Report

October 12th, 2019

Team 6

Table of Contents

Introduction **3**
- Test Purpose 3
- Methodology and Scope 3
- Summary of Results 4

Attack Narrative **4**
- 1. Employee Reconnaissance 4
- 2. Remote System Discovery 5
- 3. Vulnerability Scanning 5
- 4. File Storage Investigation 6
- 5. Website Testing 7
  - Customer Website 7
  - Crusty Croissant 9
- 6. Database Compromise 10
- 7. Privilege Escalation Attempts 10

Summary of Findings and Mitigations **11**
- Service Configuration 11
- Credentials and Authentication 12
- Database Security User Data Protection 12
- Files 13
- Website Security 14
- Compliance 15

Conclusion **16**
- Risk Rating 16
- Closing Recommendations 16
- Acknowledgements 17

Appendix **17**
- Part A - Public Employee Information Discovered 17
- Part B - Hosts and Services Discovered 19
  - Subnet 10.0.1.0/24 19
  - Subnet 10.0.2.0/24 20
  - Subnet 10.0.10.0/24 21
- Part C - Details of Vulnerabilities Found 22
- Part D - Artifacts Remaining on Machines 22

Introduction

Test Purpose

Team 6 was contracted by DinoBank, Inc. to perform a comprehensive penetration test and security audit. The purpose of the audit was to evaluate and provide security recommendations for DinoBank's applications and network infrastructure. All activities were conducted in a manner that simulated a malicious attacker engaged in a targeted attack against DinoBank, while making sure to avoid interruptions to critical business functions.

Testing began at 9am Eastern Time on October 12th, and concluded at 6pm on the same day. Due to DinoBank's status as a financial institution, emphasis was placed on identifying and exploiting security weaknesses that would allow a remote attacker to gain unauthorized access to financial data.

Methodology and Scope

The scope of the test was limited to hosts directly operating within DinoBank's network infrastructure, restricted to the five subnets specified:

- 10.0.1.0/24
- 10.0.2.0/24
- 10.0.10.0/24
- 10.0.11.0/24
- 10.0.12.0/24

To conduct the test, the team was given a collection of hosts ("jump boxes") that were already connected to DinoBank's network, and placed on the 10.0.254.0/24 network. These hosts were a combination of Kali Linux and Windows boxes, accessible over SSH and RDP respectively. Due to the scope of the network subnets, we did not evaluate wireless security, externally-facing network infrastructure (switches, routers, etc.), ATM security, cryptocurrency security, or mobile device security.

In testing the system, and due to the limited time scope and at the request of DinoBank, we focused on identifying real-world, practical attacks that an attacker might use to compromise systems. Our methodology focused on identifying a large breadth of common known vulnerabilities or misconfigurations, rather than deep and complex "zero-day" exploits.

We began our test by surveying the host network in order to identify the network topology and hosts present. We then performed comprehensive vulnerability scanning on each host we found, as well as detailed penetration tests on any web applications present. Finally, we focused on identifying weakly stored credentials, sensitive user data and files, and privilege escalation opportunities presented by each host.

Summary of Results

Initial reconnaissance of the DinoBank network resulted in the discovery of several misconfigured servers - the most important being a misconfigured FTP server and a PostgreSQL instance that both allowed access without authentication. Further investigation of the storage mechanisms found sensitive customer data being stored on the PostgreSQL server without adequate encryption techniques, as well as sensitive files stored on the FTP server. Additionally, we uncovered two DinoBank websites that contained a series of common web vulnerabilities, including lack of input validation, potential XSS, lack of HTTPS, and SQL injection, among others.

We also conducted a search on the public internet and found a variety of company data that was inadvertently made public. Finally, we identified a range of suspected remote vulnerabilities and potential privilege escalation opportunities with other hosts on the network. These results, along with many other more minor vulnerabilities, indicate a set of severe security vulnerabilities within DinoBank infrastructure. These should be rectified as quickly as possible to ensure immediate protection of customer and employee data and compliance with regulatory standards.

Attack Narrative

In this section, we enumerate at a high level the steps we took throughout the audit process. The goal is to help explain our attack methodology, provide context on our findings, and give a sense of similar steps that an attacker would take when encountering DinoBank's systems.

1. Employee Reconnaissance

We began the penetration test by evaluating what an attacker outside of DinoBank's private network would have access to, by auditing information about DinoBank available on the public internet. This allowed us to evaluate the company's public attack vectors, as well as find information that could be of use later on in the audit. This included a survey of:

- Public employee profiles across websites like LinkedIn and Facebook
- Code made publicly available by employees on sites such as Github
- Content posted by employees on sites like Twitter, Medium, Reddit, Instagram, and Pinterest
- Public DNS records and WHOIS information

While most data we found was benign, we did find examples of sensitive employee data that should not have not been made public. This data is detailed in the Summary of Findings section. In case DinoBank would like to review it later, the full list of public employee accounts we found is detailed in Appendix A.

A sample of Twitter, LinkedIn, and Facebook accounts for a DinoBank employee. Information present on these accounts can be useful reconnaissance for an attacker.

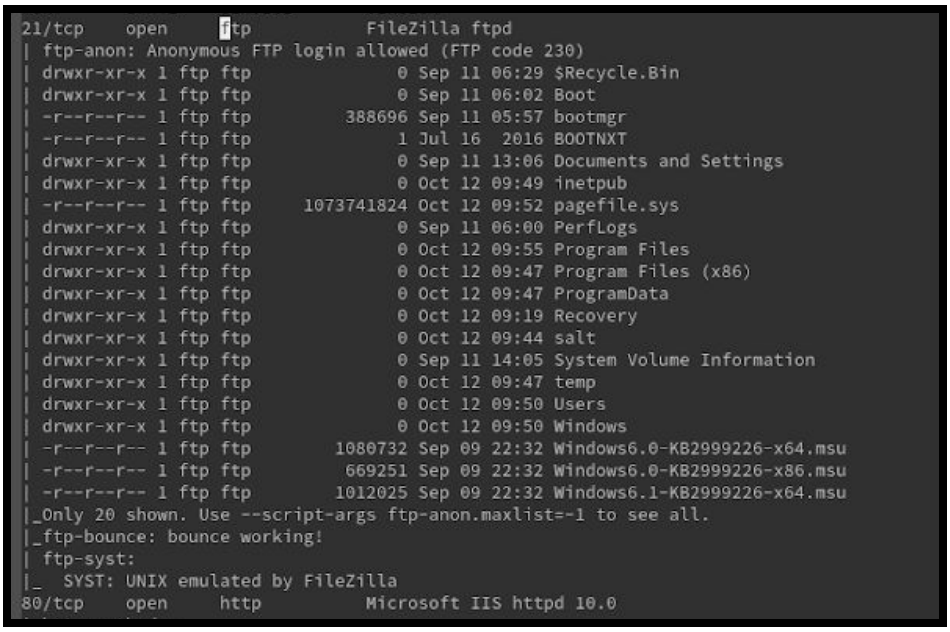
2. Remote System Discovery

Using the access given to us, we began with a network scan to enumerate live hosts on the network and services running on them. This served to inventory all of DinoBank's assets and enumerate potential machine targets, and also represents one of the first steps an attacker would take upon gaining access to the network.

Appendix B details the full list of hosts and services that we found. We were unable to directly find hosts on two of the five subnets - 10.0.11.0/24 and 10.0.12.0/24. This is likely because these hosts were only directly accessible from hosts on the other three subnets.

3. Vulnerability Scanning

For each host we found, we tested a comprehensive set of remote exploits against each of the running services. These exploits can include actions like stealing information off of the machine's running processes, using a remote buffer overflow to gain shell access, or denial-of-service attacks. The goal was to find what would typically be the "lowest-hanging fruit" for an attacker - the easiest vectors for gaining unauthorized access to a remote host.



An example of an nmap scan to find remote vulnerabilities - in this case, anonymous login allowed by the FTP server.

Some of the attacks tested included (but were not limited to):

- Running information-gathering scripts against SMB - during which we discovered several hosts had disabled SMB signing
- Trying credential reuse attacks against Windows RPC and RDP
- Attempting anonymous login to FTP and PostgreSQL - which was successful
- Attempting to brute-force SSH credentials
- Testing known CVEs for Asterisk, ServeToMe, and the Splunk daemon
- Running brute-force directory attacks against HTTP servers

A full set of the vulnerabilities we found from this step is detailed in the Summary of Findings section.

4. File Storage Investigation

As DinoBank is a financial institution, checking the network for sensitive information was an important focus of our penetration test. One of the hosts that we found in the previous step was an FTP server. Upon further investigation, we discovered that this server allows anonymous users to view files contained on it - in other words, it did not require authentication.

We discovered multiple sensitive files stored on this FTP server. For example, in `C:\salt\conf\pki\minion\minion.pem`, we found a private RSA key which would allow for spoofing the FTP server's identity, potentially allowing for gaining access to other computers on the network. Furthermore, in some places there existed plaintext passwords for services, such as in `C:\Program Files\SplunkUniversalForwarder\etc\system\default\server.conf`,

which presents opportunities for further attack vectors. The full set of sensitive data found is detailed in the Summary of Findings section.

The RSA private key that was found in C:\salt\conf\pki\minion\minion.pem (sensitive info has been redacted).

5. Website Testing

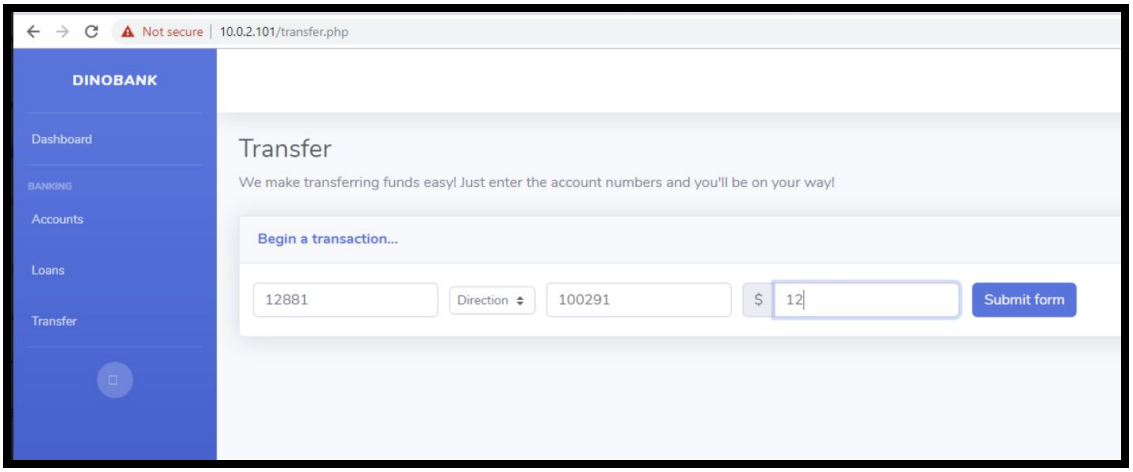
We also discovered several DinoBank websites running on the hosts, which we ran penetration tests against.

Customer Website

One host was running the DinoBank customer website, which lets customers create bank accounts, apply for loans, or transfer funds between account of DinoBank customers.

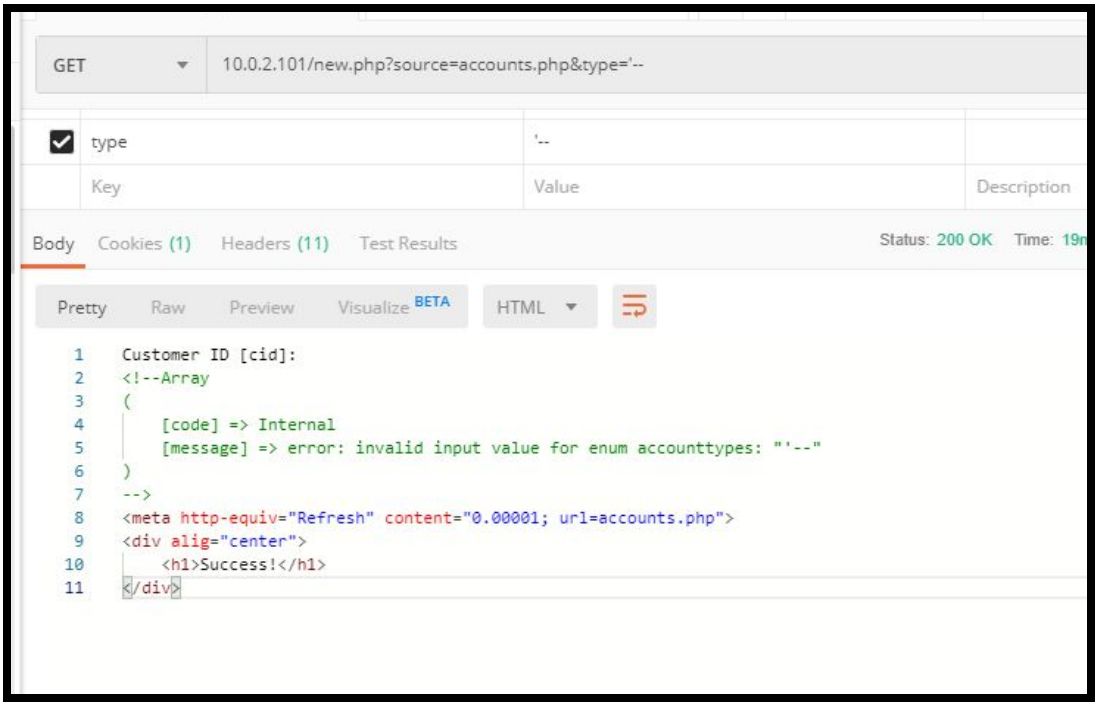
We began by attempting to create an account on the site, which revealed that the input was not sanitized - any set of personal information, such as names with invalid characters or invalid email

addresses, would be accepted by the website. Moreover, the website allowed for creating bank accounts without any sort of verification process of customer data, such as address or SSN.



A screenshot of the transfer section of the DinoBank website. Any user can transfer money between any set of accounts, provided they know the account IDs.

After creating an account, we found that the bank's transfer UI allowed any user to transfer money to themselves from another account without consent, provided they know the other user's account number. We were able to verify this by transferring $1 from a random account to the account we just created (the dollar was transferred back afterwards), using account IDs gathered from the database compromise (detailed below).

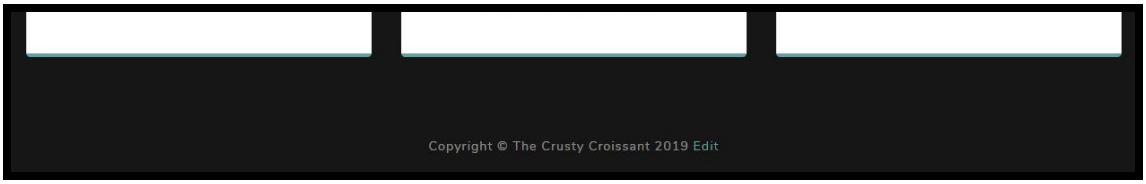


An example request sent to the DinoBank server. Note how the comment of the response leaks information about the SQL table.

We also noticed that many pages have errors that leak information about the state of the website. For example, if an error occurs while creating a user, the error message includes the title of the column in the database where the error occurred. This information could be used by attackers to craft an exploit against the database containing login information. We also suspected that this means the website is vulnerable to SQL injection. We ran the sqlmap tool against it, but did not find any direct injections.

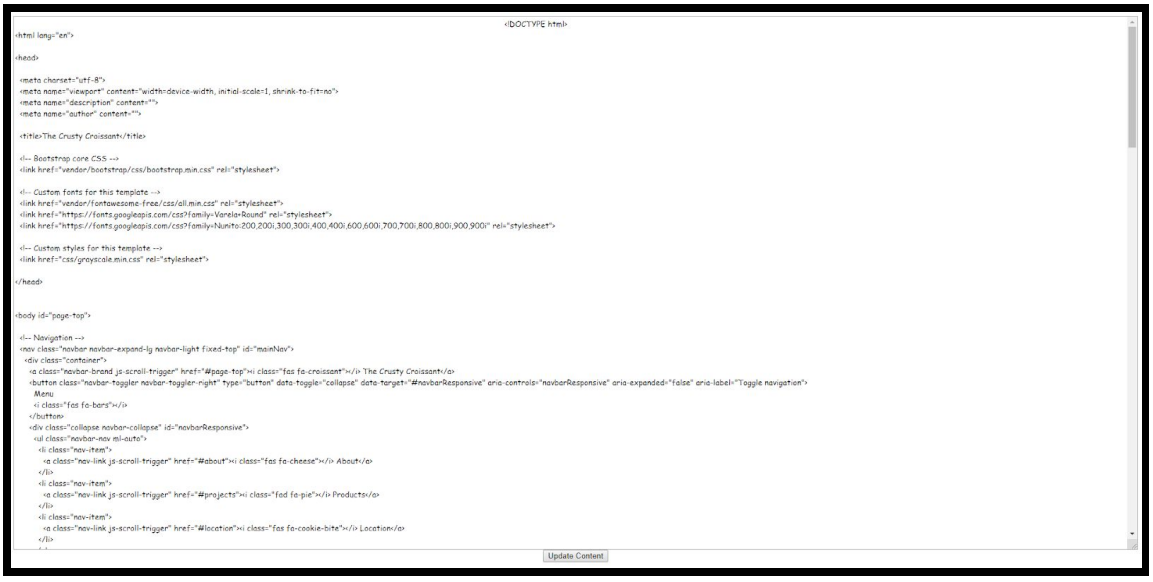
Crusty Croissant

Another website found was the landing page for the Crusty Croissant (also referred to as Crispy Croissant), a cafe located in one of the Dino Bank offices. At the bottom of the page, we found an 'Edit' button that allowed any user to directly edit the source code of the website.



The "Edit page" link on the bottom of the Crusty Croissant website.

This input field was vulnerable to XSS payloads, which could introduce unwanted risks to customers that visit the Crusty Croissant website. We were able to verify that the XSS vulnerability was viable using a polyglot payload.



An example XSS payload embedded in the website source.

More details about other website vulnerabilities we found are included in the Summary of Findings section.

6. Database Compromise

We had also identified a PostgreSQL database on the network. This server was vulnerable to schema enumeration and hash-dumping techniques - but more importantly, we discovered it did not require authentication. This meant that any individual on the network could connect to the server.

Moreover, we discovered that this table contained information about all DinoBank employees, users, and bank accounts - meaning any accessor could read and write account balances and transactions. We also discovered that user credentials were written in plaintext - meaning the attacker could reuse these credentials to hijack user or employee accounts. Additional sensitive customer data was stored in this database, including job titles, phone numbers, email addresses, social security numbers, and street addresses.

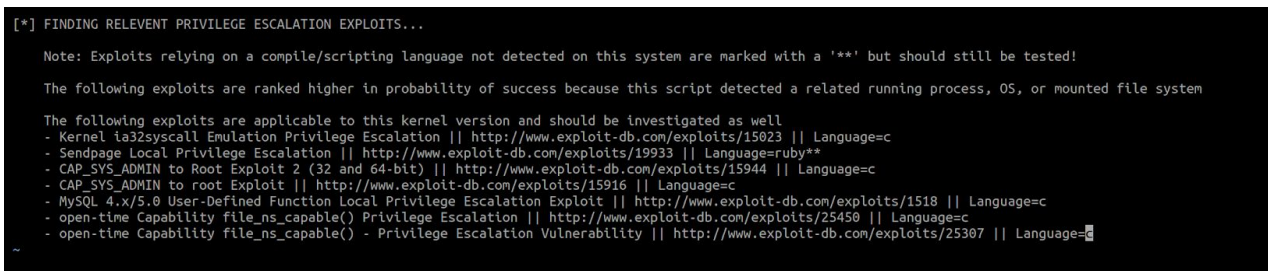
A screenshot of the database users table. Sensitive information has been redacted.

Additionally, by accessing the account associated with the PostgreSQL database, we were able to create a remote shell - allowing us to run arbitrary commands on the machine. This allowed us to read files on the machine, which included the full source of the DinoBank API that was also running on the machine.

We found a variety of other issues present on the database server and API, which are detailed in the Summary of Findings section.

7. Privilege Escalation Attempts

Using the aforementioned PostgreSQL machine, we attempted to escalate privileges in order to gain access to a root (“Administrator”) account, as a survey script indicated that the machine was vulnerable to several privilege escalation exploits.



Results of the privilege escalation survey script.

We tested several of these exploits that corresponded to the kernel version, including ia32syscall emulation privilege escalation and CAP_SYS_ADMIN to Root. We confirmed that the machine is not vulnerable to either of these attacks. However, we believe that the machine may be vulnerable to CVE 2018-18955 (detailed in Appendix C).

Summary of Findings and Mitigations

The following tables detail each vulnerability that we found across DinoBank's networks, as well as the associated risk factor and recommended mitigation. We have organized the findings into a set of categories to simplify parsing.

Service Configuration

Title	Risk Level (low/medium/high)	Description
No auth required for FTP server	high	The FTP server present on the network allows anonymous (unauthenticated) access. This would allow a malicious adversary to view shared files without a login.
Vulnerable to SlowLoris (DoS/DDoS)	high	Multiple HTTP servers are vulnerable to SlowLoris attacks, which are a form of low-bandwidth DoS/DDoS. This attack allows for a relatively small number of clients to exhaust thread resources on servers. Updating the responsible services will mitigate this threat. In the event that no update is available, it is recommended to implement DoS/DDoS mitigation services such as CloudFlare.
SMB signing disabled or not required	high	Most of the Windows machines have been misconfigured such that SMB signing is either not required or disabled entirely. This represents a potentially serious misconfiguration, as SMB signing is designed to protect against man-in-the-middle attacks. Mitigating this vulnerability is as simple as setting a Group Policy that requires SMB signing for all domain-joined machines.
Unnecessary services running on machines	medium	Many of the machines on the network have services running that we suspect are not essential for DinoBank's core functionality. In addition to being more costly to maintain, redundant services present a greater attack surface for a potential attacker. For example, one Windows Group Policy prevents Remote Desktop for hosts, but those hosts still have the RDP port open. Appendix

		B details all the running services for each host, and annotates the ones that we suspect may not need to run.
FTP Bounce	medium	The PORT command is enabled on the FTP service. This would allow an attacker to use FTP to scan other machines from the same subnet as the FTP machine, allowing access to machines the attacker would not have had direct access to otherwise.
Time Sync	low	Several hosts have different time zones set from other hosts on the network. Time should be synchronized between all computers in the same timezone in order to better facilitate incident response (such as evaluating log files).

Credentials and Authentication

Title	Risk Level (low/medium/high)	Description
Employee credentials revealed publicly	high	A survey of publicly-available information turned up public and private SSH keys on the DinoBank Github repo. This information should be scrubbed from the commit history.
Employee credentials stored in plaintext	high	The database stores unhashed, unsalted employee credentials. These should be hashed and salted to prevent an attacker from being able to easily reuse these in case of a database compromise.
Weak/ weakly enforced password policy	high	An audit of the credentials stored in the database shows that the current password policy enforced is weak. Simple passwords, such as “Password1!”, are in use by several employees and users. These passwords are easy to brute-force by an attacker.
API Tokens Stored in Source Code	medium	The source code of the API has the access tokens embedded in the source code. These tokens should be stored as environment variables, so that an attacker would not be able to gain access to these tokens if the source code were compromised.

Database Security User Data Protection

Title	Risk Level (low/medium/high)	Description

Passwords not required	high	The PostgreSQL database does not require passwords to log into the bank user or postgres user. This gives attackers remote access to all customer and employee data.
User credentials stored in plaintext	high	The database has unsalted, unhashed user passwords stored. This allows attackers to steal passwords for reuse, which not only compromises services hosted by Dinobank, but also compromises any services where a customer uses duplicate passwords. These should be hashed and salted to prevent an attacker from being able to easily reuse these in case of a database compromise.
Sensitive customer info stored without isolation	high	Extremely sensitive customer data is stored in a database that has minimal authentication controls. This data includes social security numbers, phone numbers, email addresses and street addresses. Ideally, this database should be quarantined from the rest of the network.
Customer and Employee database mixed	low	Employee data and customer data should be stored on separate machines. This prevents unauthorized access to one of these credential types from resulting in a compromise of the other.

Files

Title	Risk Level (low/medium/high)	Description
Private RSA Keys	high	Private RSA keys were found in on the FTP server `C:\salt\conf\pki\minion\minion.pem`. These could be used to authenticate elsewhere on the network and lead to privilege escalation.
Default/ plaintext Passwords	medium	Services such as Splunk had plaintext passwords in use, such as in `C:\Program Files\SplunkUniversalForwarder\etc\system\default\server.conf`. These can be used to gain access to the services on the system and, like the RSA keys, can lead to privilege escalation.
Updates staged on FTP server	low	Files for Windows updates were staged on the FTP server. This potentially indicates that Windows updates are not being automatically rolled out, and that updates are being loaded from

	the FTP server. If this is true, an attacker could modify the files on the FTP server to load modified versions of Windows onto the Windows hosts.

Website Security

Title	Risk Level (low/medium/high)	Description
Unauthorized transfers	high	The DinoBank customer website allows a customer to transfer money from any account to any other account, provided they know (or guess) the account IDs. This can be done through the website's transfer page, and does not require confirmation from either party.
HTTP instead of HTTPS	high	Of the websites we surveyed, none made use of HTTPS. This means that if an attacker were on the same network as the user, they could survey the user's communication with the website - including login credentials and financial transactions. We recommend using HTTPS (SSL/TLS) on all DinoBank websites.
Editable website contents	high	The "Crusty Croissant" website has an option to edit the contents of the page. This would allow an attacker to edit the page to add false information or a cross-site-scripting (XSS) payload. Origin pollution is also possible, meaning that malicious scripts from anywhere on the internet could be run within the DinoBank origin and therefore bypass the security granted by the same-origin policy.
No HTTPOnly cookies	medium	The DinoBank customer website does not make use of the http-only header. This header forces page cookies to only be used in HTTP requests, thus making them inaccessible to JavaScript. Without this flag, JavaScript injected into the page (for example through XSS) could access the user's session cookie, making it easier for an attacker to steal user sessions.
SQL database info leak	medium	The login, registration, and account transaction pages (register.php, alert.php, new.php) leak information about the SQL database during queries. This could allow an attacker to gain information about the database schema, and potentially make it easier to perform SQL injections.
Customer information	medium	When logging in with an invalid username, the website responds with "invalid user". This can be used to leak

leak		information about who DinoBank's customers are, violating user privacy. Instead, the website should report something like "the user credentials were invalid" and not reveal if the user exists in the database or not. Additionally, the new.php page leaks the customer ID upon creating a transaction.
Various bugs	low	For example, the "Already have account?" button redirects to an invalid page, and the site tries to load JavaScript and CSS files that do not exist. These are not security vulnerabilities in and of themselves, but indicate the presence of other bugs that could be vulnerabilities.

Compliance

As a leading financial institution, DinoBank should maintain compliance with several regulatory frameworks and agencies - including GLBA, PCI, FinCen, KYC/AML, FINRA, and others. This penetration test does not serve as a full audit of compliance within all of these regulatory frameworks. However, many of these regulatory frameworks specify strict standards for data encryption, data collection consent and secure storage, which our security audit has revealed some insight into.

For example, the Social Security Administration recommends encryption of social security numbers (SSNs). Additionally eleven states have legislation prohibiting the transfer of social security numbers over unencrypted channels[1]. DinoBank may be in violation of these regulations, since SSNs are stored unencrypted in the database and all web traffic is transmitted over HTTP.

Additionally, the Gramm-Leach-Bliley Act is a United States federal law that requires that banks protect personal information. DinoBank may not be currently GLBA compliant, because sensitive personal data is unprotected from unauthorized access - there are 8,855 instances of sensitive data accessible on DinoBank networks. As a result, DinoBank risks being fined $100,000 per violation, and individuals found guilty of violations risk a prison sentence of up to 5 years[2].

[1] Social Security Numbers: Federal and State Laws Restrict Use of SSNs, yet Gaps Remain https://www.govinfo.gov/content/pkg/GAOREPORTS-GAO-05-1016T/html/GAOREPORTS-GAO-05-1016T.htm

[2] Data Insider: What is GLBA Compliance? Understanding the data protection requirements of the Gramm-Leach-Bliley Act https://digitalguardian.com/blog/what-glba-compliance-understanding-data-protection-requirements-gramm-leach-bliley-act

Finally, because DinoBank does not verify social security numbers before allowing a user to open a bank account, they may be in violation of the Patriot Act[3], as documentation proving that a social security number is valid is required to open a bank account.

Again, this audit does not serve as a definitive statement on DinoBank's compliance (or lack thereof), but we recommend immediate investigation into each regulation.

Conclusion

Risk Rating

The overall risk identified to DinoBank as a result of this penetration test is **high**, as direct paths from initial network access to full data compromise have been discovered. It is reasonable to believe that a malicious entity would be able to successfully execute such attacks against DinoBank, and we recommend that remediation efforts are undertaken immediately.

Closing Recommendations

While the previous section enumerates the specific set of vulnerabilities and mitigations we found, we recommend the following high-level practices:

- Minimize attack surface exposure by limiting unnecessary services. While our limited security audit cannot infer the full extent of which services are strictly necessary,, we have included a recommendation of services we believe may be redundant in Appendix B.
- Enforce authentication between all services. As a general rule of thumb, no service containing sensitive data or having privileged access should be left unauthenticated. Moreover, all services should enforce the principle of least privilege. In other words, employees should not have access to services that lie outside of their core job function.
- Review credentials in use at the company - in particular, migrate databases to make use of hashed and salted passwords. Enforce password policies for both employees and users, and regular password rotations for employees. Make sure access tokens and SSH keys are not accidentally included in code or committed to Github.
- Thoroughly audit the DinoBank website source code, and ensure that modern web security practices are being followed - including but not limited to HTTPS, input sanitization, XSS protection, and user verification.
- Comprehensively review the set of files stored on servers across the company. Make sure sensitive info is only stored in places where it is strictly required.

[3] USA PATRIOT Act Regulations
https://www.csiweb.com/industries-we-serve/financial-institutions/regulatory-compliance/federal-regulations/patriot-act-compliance

- Create a set of policies based on the above recommendations, and schedule regular assessments to verify they are being followed in the future.

Finally, Appendix D contains a short list of artifacts left behind on the servers during the penetration test, in case DinoBank would like to identify these files later.

Acknowledgements

Finally, we would like to thank DinoBank for choosing to enlist our firm for this security audit, and for the clear and consistent communication throughout. We know you have many penetration specialists to choose from, and we appreciate the opportunity to work with your company. We hope to work with you all again in the future.

Appendix

Part A - Public Employee Information Discovered

This list details a list of employees with public information and corresponding links. While not all of these are sensitive company data, it serves as a good starting point for auditing public DinoBank information.

Employee	URL
Dan Oliver	https://github.com/DinoDanOliver https://www.reddit.com/user/dino_dan_oliver https://www.pinterest.com/dinodanoliver/ https://dino-dan-oliver.tumblr.com https://www.linkedin.com/in/dan-oliver-98a942191/
Megan	https://github.com/dinomegan https://www.reddit.com/user/dinomegan https://www.tumblr.com/blog/dinomegan
Slade Hunter	https://github.com/DinoSladeHunter https://www.reddit.com/user/DinoSladeHunter https://www.pinterest.com/dinosladehunter/ https://www.tumblr.com/blog/dinosladehunter
Cale Strickland	https://github.com/DinoCale https://www.reddit.com/user/DinoCale https://www.pinterest.com/dinocale/ https://www.tumblr.com/blog/dinocale https://www.linkedin.com/in/cale-strickland-b89947191/
Alex Woods	https://www.reddit.com/user/dinoalexwoods/

Heather Potter	https://github.com/DinoPotter https://www.reddit.com/user/DinoHeatherPotter/ https://www.linkedin.com/in/heather-potter-b8290b191/
Alex Faulkner	https://medium.com/@dino.alex.faulkner https://www.reddit.com/user/dino-alex-faulkner/ https://www.linkedin.com/in/alex-faulkner-343509192/ https://twitter.com/AlexFaulkner17 https://dino-alex-faulkner.tumblr.com/
Dahlia Dawson	https://www.facebook.com/dahlia.dawson.357 https://www.reddit.com/user/dahlia-dawson https://www.linkedin.com/in/dahlia-dawson-06600a192/ https://twitter.com/dawlia7
Brennan Easton	https://www.facebook.com/easton.brennan.98 https://www.reddit.com/user/brennan-easton https://www.linkedin.com/in/easton-brennan-6b600b192/ https://twitter.com/brennan_easton
Abril Reyess	https://www.facebook.com/abril.reyess.129 https://www.linkedin.com/in/abril-reyess-ab04b2192/ https://twitter.com/ReyessAbril
Meredith Sournoise	https://www.reddit.com/user/merefromthebank/ https://www.instagram.com/merefromthebank/ https://www.linkedin.com/in/meredithfrormthebank/ https://twitter.com/merefromthebank
Mitchell Zamora	https://www.linkedin.com/in/mitchell-zamora-0a150a192/
Jacqueline Woods	https://www.linkedin.com/in/jacqueline-woods-715933192/
Lawrence Hayden	https://www.linkedin.com/in/lawrence-hayden-161504192/
Jamie Davenport	https://www.linkedin.com/in/alex-faulkner-343509192/
Paul Alvarado	https://www.linkedin.com/in/paul-alvarado-5308b7192/
Ruth Brooks	https://www.linkedin.com/in/ruth-brooks-65700b192/ https://twitter.com/RuthBro60237251
Precious Braun	https://www.linkedin.com/in/precious-braun-5144b2192/
Tom Dickson	https://www.linkedin.com/in/tom-dickson-52a0b0193/
Mauren Davenport	https://www.linkedin.com/in/mauren-davenport-62500a192/
Johnathan Gay	https://www.linkedin.com/in/johnathan-gay-8a2066192/

https://twitter.com/Jonatha43789365

Ariel Robinson https://www.linkedin.com/in/ariel-robinson-154b50192/

Part B - Hosts and Services Discovered

The following tables detail the hosts and services found on the DinoBank network. Underlined services denote services we believe may be redundant, and could be turned off to minimize potential attack vectors - though please note that these are estimates, and that these services should be tested for use before being disabled.

Subnet 10.0.1.0/24

IP	OS	Machine Purpose	Ports and Services Found
10.0.1.10	Windows Server 2016 Datacenter 6.3	Domain Controller	**53** - DNS, **88** - Kerberos, **135, 49664, 49665, 49666, 49668, 49671, 49676, 49678, 49682, 49694, 55491** - RPC, **139** - Netbios, **389** - LDAP, **445** - Directory Services, **464** - kpasswd, **593, 49675** - RPC over HTTP 1.0, **636** - LDAP over TLS/SSL, **3268** - AD LDAP, **3269** - Global Catalog over SSL, **3389** - Remote Desktop, **5985, 5986, 47001** - HTTPAPI 2.0, **9389** - .NET Message Framing, **9971** - ServeToMe
10.0.1.11	Windows		**135, 49664, 49665, 49670, 49675, 49690, 49693, 49695, 49718** - RCP, **139** - Netbios, **445** - Directory Services, **3389** - Remote Desktop, **5985, 5986, 47001** - HTTPAPI 2.0, **8089** - Splunkd, **9971** - ServeToMe
10.0.1.12		File Server, Web Server	**21** - FTP, **80** - HTTP, **135, 49664**, **49665**, **49669**, **49670**, **49690**, **49693**, **49712**, **49716** - RPC, **139** - Netbios, **445** - Directory Services,

		3389 - Remote Desktop, **5985, 5986, 47001** - HTTPAPI 2.0**,** **8530** - HTTPD 10.0 (IIS), **8531** - Unknown, **9971** - ServeToMe
10.0.1.20	Mail Server, Web Server	**25, 465, 476, 477, 587, 717, 2525** - SMTP, **80, 81** - HTTP, **135, 2103, 2105, 2107, 6400, 6401, 6402, 6405, 6424, 6430, 6433, 6469, 6473, 6488, 6519, 6533, 6537, 6538, 6543, 6552, 6562, 6569, 6570, 6577, 6582, 6584, 6585, 6593, 6598, 6617, 6618, 6621, 6645, 6653, 6679, 6708, 6742, 6780, 6824, 6896, 6925, 6935, 6963, 6965, 6970, 6990, 7034, 7067, 7102, 7201, 7253, 7774** - RPC, **139** - Netbios, **443** - HTTPS, **444, 8172** - HTTPD 10.0 (IIS), **445** - Directory Services, **593, 6001** - RPC over HTTP 1.0, **808, 5060, 5062, 5065, 61294** - Unknown, **890, 3801, 3803, 3823, 3828, 3843, 3863, 3867, 9710, 64337** - .NET Message Framing, **3389** - Remote Desktop, **3800, 5985, 5986, 47001** - HTTPAPI 2.0, **3875, 64327** - Microsoft Exchange 2010 Log Copier, **9971** - ServeToMe
10.0.1.33	Web Server	**22** - SSH, **80** - HTTP, **9971** - ServeToMe
10.0.1.50		**135, 49664, 49665, 49671, 49672, 49689, 49700, 49711, 49723** - RPC, **139** - Netbios, **445** - Directory Services, **3389** - Remote Desktop, **5985, 5986, 47001** - HTTPAPI 2.0, **9971** - ServeToMe
10.0.1.250		**OUT OF SCOPE**

Subnet 10.0.2.0/24

IP	OS	Machine Purpose	Services and Ports Found

10.0.2.100	Linux	Database for Website	**22 -** SSH **80** - HTTP/dinobank API, **5432 -** PostgreSQL, **3389 -** dinobank API
10.0.2.101	Linux	Web Server	**22 -** SSH, **80** - HTTP
10.0.2.102	Linux	Work Station	**22** - SSH
10.0.2.103	Linux	Work Station	**22** - SSH, **80** - HTTP/QueryTree
10.0.2.200	Linux	Work Station	**22** - SSH

Subnet 10.0.10.0/24

IP	OS	Machine Purpose	Services and Ports Found
10.0.10.5	Ubuntu	Web Server	**22** - SSH (OpenSSH), **80** - HTTP (Apache2)
10.0.10.100	Windows	Work Station	**135** - RPC, **139** - Netbios, **445** - Directory Services, **3389** - Remote Desktop, **5985, 5986, 47001** - Windows Remote Management, **9971** - StreamToMe
10.0.10.201	Windows	Work Station	**135** - RPC, **139** - Netbios, **445** - Directory Services, **3389** - Remote Desktop, **5985, 5986, 47001** - Windows Remote Management, **9971** - StreamToMe
10.0.10.202	Windows	Work Station	**135** - RPC, **139** - Netbios, **445** - Directory Services, **3389** - Remote Desktop, **5985, 5986, 47001** - Windows Remote Management, **8089** - Splunkd, **9971** - StreamToMe

10.0.10.209	Windows	Work Station	**135** - RPC, **139** - Netbios, **445** - Directory Services, **3389** - Remote Desktop, **5985, 5986, 47001** - Windows Remote Management, **9971** - StreamToMe

Part C - Details of Vulnerabilities Found

Sensitive Public Information Discovered

- https://github.com/DinoDanOliver/.files
  - SSH Public Key
    - https://github.com/DinoDanOliver/.files/blob/c39e404d2905b3f6f0dd4c034ea0f810b5f2ab16/files/ssh/dinodan
  - SSH Private Key
    - https://github.com/DinoDanOliver/.files/blob/c39e404d2905b3f6f0dd4c034ea0f810b5f2ab16/files/ssh/dinodan

CVE-2018-18955 - Possible Privilege Escalation

- Linux kernels 4.15.0 to 4.18.18
- https://www.exploit-db.com/exploits/45915

CVE-1999-0017 Detail - FTP bounce server

- FTP Server with Port Command enabled
- https://www.cvedetails.com/cve/CVE-1999-0017/

CVE-2018-17189 - Slowloris

- Denial of Service in HTTP servers
- https://www.cvedetails.com/cve/CVE-2018-17189/

Part D - Artifacts Remaining on Machines

Scripts remaining on PostgreSQL machine (10.0.2.100)

- `linuxprivchecker.py` in `/var/lib/postgresql` directory
- `linuxprivchecker.py`, `a.out`, `a.out.1`, and `.so` in `/tmp` directory